AFRL-RI-RS-TR-2012-080

Analysis and Design of Complex Network Environments

BRIGHAM YOUNG UNIVERSITY

MARCH 2012

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2012-080 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/
ROBERT L. KAMINSKI
Work Unit Manager

/s/
WARREN H. DEBANY, JR., Technical Advisor
Information Exploitation & Operations Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
MAR 2012	Final Technical Report	SEP 2009 – SEP 2011

4. TITLE AND SUBTITLE

Analysis and Design of Complex Network Environments

5a. CONTRACT NUMBER
FA8750-09-2-0219

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Sean Warnick
Daniel Zappala

5d. PROJECT NUMBER
UGOV

5e. TASK NUMBER
BY

5f. WORK UNIT NUMBER
U1

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Brigham Young University
Office of Research & Creative Activities
Dept of Computer Science
Provo, UT 84602-1231

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/Information Directorate
Rome Research Site/RIG
525 Brooks Road
Rome NY 13441

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2012-080

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This research takes a fresh view of "complex" network environments and answers fundamental questions about 1) how to model them, 2) the design of experiments necessary to discover their structure (and thus adapt system inputs to optimize the resulting performance), and 3) the relationship between network structure to vulnerability and attack. Specifically, this work explores these issues in the context of both wireless mesh and bio-chemical reaction networks. Although wildly different application areas, this research unites theoretical work that clarifies fundamental limitations of complex networks with network engineering and systems biology to implement specific designs and experimentally verify the theoretical discoveries in what is being called the "network design cycle." The network design cycle iterates through a process of mathematical modeling, problem formulation, algorithm design, implementation, and experimental validation that drives the development of practical theory. Using this process this effort demonstrated new models and rate control protocols for wireless mesh networks, along with experimental validation.

15. SUBJECT TERMS

Complex Networks, Network Design, Wireless Mesh Networks, Bio-chemical Reaction Networks

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			ROBERT L. KAMINSKI
U	U	U	UU	103	19b. TELEPHONE NUMBER (*Include area code*) N/A

Table of Contents

List of Figures

1 Summary

This research considers the mathematical representation of complex network environments to enable network reconstruction and analysis. In this study, a complex network environment is the interconnection of controlled dynamical systems resulting in both nonlinear and noisy behavior as well as complicated interconnection structure. Classical representations of such systems, such as coupled differential equations, are not effective for our purposes because they are too detailed, resulting in a heavy information load on the experimental data required for reconstruction. Moreover, the typical simplification found by grouping parts of the system into well-characterized subsystems and then restricting the desired structural information to be only the interconnection structure among subsystems is equally problematic, since even here one must determine how to partition all system states (even the unmeasured ones) into the right subsystems; this can be a tremendous informational burden or even impossible.

Here, we leverage a new idea about how to represent system structure to reduce the information burden on the experimental data required for network reconstruction. In this new representation, we look for the open-loop causal dependencies among measured variables. As a result, distinct dependencies between different variables may actually depend on the same underlying unmeasured variable and thus be entangled. In this situation, this new representation would not distinguish between entangled components and non-entangled components. Nevertheless, the benefit of remaining agnostic about the potential entanglement of system dynamics through unmeasured variables is that this new type of structure can be derived from O(n) experiments, where n is the number of measured variables. Algorithms for accomplishing this reconstruction are discussed, with special emphasis on how to handle noise and underlying nonlinearities. Also, further theoretical analysis reveals interesting characteristics about minimal representations of these new models and their potential for conducting structure-dependent robustness analyses. This robustness analysis provides the basis for exploring vulnerability and network security.

A unique strength of this work is its equal effort on both theoretical development and experimental validation. Using wireless mesh networks as an application testbed, the "network design cycle" is employed to iteratively develop theoretical models, rate control protocols, and experimental validation in a process that lays the foundation for the development of a model of wireless mesh networks suitable for the new network reconstruction methods. Although not yet complete, this work launches new possibilities for wireless mesh networks where system intruders can be automatically detected by the way they induce coupling among link rates in the network. This "packet-free" approach is distinct from current techniques and suggests a bold new research program. Bio-chemical reaction networks provide another testbed for theoretical validation, and that work is just emerging.

2 Introduction

The study of network environments have received considerable attention from various disciplines over the last 20 years. This is in part due to the rise of the internet, but it also is due to advances in biology that have brought new kinds of networked "machines" into clear view that contrast sharply with our solid-state, engineered systems. The research discussed here

takes a fresh view of these new, "complex" network environments and answers fundamental questions about 1) how to model them, 2) the design of experiments necessary to discover their structure (and thus adapt system inputs to optimize the resulting performance), and 3) the relationship between network structure to vulnerability and attack.

Specifically, this work explores these issues in the context of both wireless mesh and biochemical reaction networks. Although wildly different application areas, this research unites theoretical work that clarifies fundamental limitations of complex networks with network engineering and systems biology to implement specific designs and experimentally verify the theoretical discoveries in what we call the "network design cycle."

2.1 What is a "complex" networked environment?

The descriptor "complex" has been used in other studies to characterize networks that are large, meaning that a graph used to represent the network has a large number of nodes. Often the number of edges is also taken to be large and devoid of certain regular structure, so that, for example, a tree-structured graph would not be considered "complex." In these studies one typically looks for hidden regularities or patterns that characterize the structure of the graph in simple terms in spite of these other "complexities." A typical example would be small-world networks.

In this work, we consider different environments, where "network" refers to an interconnected dynamical system. In these environments, system behavior is as important a characteristic of the network as is its structure. The descriptor "complex" then refers to both the network behavior as well as its structure.

This work addresses networks with complex behavior by considering systems with underlying nonlinear dynamics and noisy measurements. As a first step, the results presented here leverage Lyapunov theory to apply linear analysis locally to regions near equilibria of the underlying nonlinear system. Extending these results globally to the underlying nonlinear system is not immediately obvious and requires new thinking about the meaning of structure, beyond that already developed in this research. As a result, global analysis is left for future research.

This work also addresses networks with complex structure. Besides the usual definition, of a "complex" network structure represented by a graph with a large number of nodes and non-trivial edge patterns (e.g. allowing for arbitrarily complicated feedback relationships), this work also makes a particular contribution by developing representation and analysis tools applicable to networks of dynamical systems with potential hidden entanglements among unmeasured variables. This "potential entanglement" type of network complexity is previously unaddressed in the literature, yet it becomes particularly important for inferring network structure from behavioral data.

Appreciating the power of structural representations that allow for potential entanglement among unmeasured variables to simplify network inference problems is subtle, but it is a central contribution of this research program. Consider, for example, a network that is composed of the interconnection of various subsystems. By definition, each subsystem's internal states only affect that subsystem, and the interconnection variables are themselves measured quantities. Inferring this network "subsystem" interconnection structure from data thus demands the discovery of the true partition of all of the system's *unmeasured* states

into their appropriate subsystem components. Discovering such a partition from measured data can be extremely difficult or impossible. This research, on the other hand, leverages a different representation of system structure, called signal structure, that does not rely on the idea of subsystems and allows for potential entanglement among unmeasured states. As a result, inferring a system's signal structure requires much less information, and thus fewer experiments, than inferring a system's subsystem structure.

One contribution of this research is to thoroughly understand the relationships between a system's subsystem and signal structures. Often, systems with solid-state components (such as routers in the internet) have subsystem and signal structures that are equivalent. Sometimes, however, systems have a fluid-like character (such as bio-chemical reaction networks or wireless mesh networks) and the resulting subsystem and signal structures can be very different. Characterizing informativity conditions and developing scalable algorithms for network reconstruction (i.e. inference) of signal structure are the primary theoretical contributions of this effort. These models of complex networked environments also facilitate a novel robustness analysis that leads to new results about system vulnerability and security. These contributions are highlighted when applied to complex network environments that exhibit both the behavioral and structural complexity as described.

2.2 What is the network design cycle?

Besides the theoretical contributions of this work, this research represents an active collaboration between theoretical development and physical implementation and testing on real complex networks. This collaboration is most evident in our work on wireless mesh networks, where active modeling and protocol design efforts lead to simulation, implementation, and experimental testing on our live wireless mesh testbed. Similar collaborations for bio-chemical reaction networks began to emerge during this study, but the implementation and experimental testing is incomplete and part of ongoing research.

The network design cycle defines the scientific process we engage that unites our theoretical and applied work. The next section discusses each part of the design cycle in detail as part of our research methods, assumptions, and procedures. Section 7 the offers highlights of the research program results, beginning with specific applications and ending with general theoretical results. Sections 7.1 and 7.2 detail our development of new models for wireless mesh networks, leading to new rate control protocols that can drastically improve network performance. Section 7.3 and 7.4 then demonstrate our network reconstruction efforts applied to nonlinear bio-chemical reaction networks with noisy measurements. Section 7.5 then highlights the general theoretical underpinnings regarding the meaning of structure in interconnected dynamical systems and the mathematical relationships among different types of structure. Section 8 then concludes the report.

3 Methods, Assumptions, and Procedures

To address these problems, our work uses a method we call the Network Design Cycle, as shown in Figure 1. We start by formulating a mathematical model of the network, precisely characterizing how it operates. In a wireless network, this may involve describing how nodes

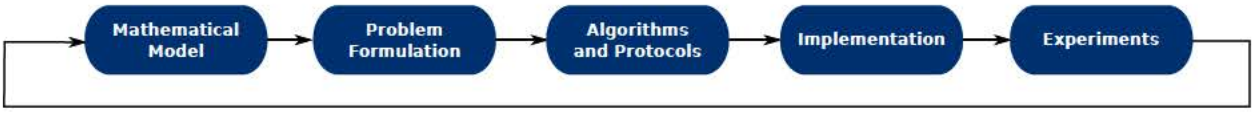


Figure 1: The Network Design Cycle

interact via carrier sensing and interference. Based on this model, we carefully formulate the research problem, such as providing optimal performance or detecting malicious nodes. The next step is to design algorithms and protocols that will solve the problem. We then implement the solution, which often requires considering additional complexities that arise when building and deploying code on a network. Finally, we run experiments to validate that the solution works as designed, for example by providing optimal performance or security guarantees. If performance deviates from what is expected, this means that our original model or problem formulation was wrong. This leads to another iteration of the design cycle, where we create a refined model or reformulate the problem.

For the steps that involve implementation and experimental evaluation, we use a variety of tools. We use MATLAB to numerically evaluate algorithms, ensuring that they converge to the expected solution and, in some cases, provide optimal performance. For experimental testing, we use a wireless mesh network deployed in our department's building, consisting of 30+ computers configured as wireless routers, spread over two floors. The mesh network can be configured so that it uses 802.11a, and we ensure that we select a channel with no other traffic, so that we can run experiments without any outside noise when desired. Our protocols are implemented using WiFu [4], a toolkit developed with funding from NSF that enables rapid deployment of experimental wireless protocols in user-space.

4 Results and Discussion

4.1 Modeling Wireless Networks with Partial Interference

Because wireless networks use shared communication channels, contention and interference can significantly degrade throughput and fairness. A common approach to solving this problem is to form a model representing the wireless network that imposes constraints on the rates, and then design the controller to compute a distributed algorithm that solves an optimization problem. In wireless networks, these constraints are primarily imposed by the channel capacity among competing nodes [6]. By formulating the problem so that it is convex, well established techniques can, in most cases, be applied to design a distributed algorithm that computes the rates [13].

Traditional graph-based models impose resource constraints using a contention graph, where communication links between nodes are represented by vertices and contention is represented by edges. Two links are said to contend with each other if they cannot be active at the same time without causing collisions. Existing graph-based models conservatively characterize interference as a binary effect [5, 10]. Under this model, an interfering node is assumed to corrupt all of the packets received at a remote node, while non-interfering nodes have no effect. Binary interference is represented in a contention graph by simply treating

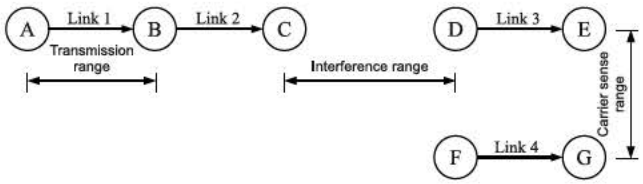


Figure 2: A network topology graph.

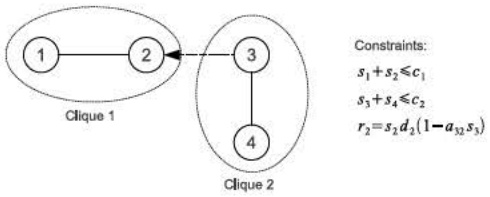


Figure 3: A contention graph for the sample network topology, based on the partial interference model.

it as contention, that is, if one link interferes with another, neither may send at the same time.

4.1.1 Model

While this approach does lead to convex problem formulations, it does not accurately represent the interference relationship among links. Contention occurs when sending nodes can sense one another and thus take turns, whereas interference occurs when two *non-contending* senders transmit simultaneously, causing a packet to be corrupted at a receiving node. A recent measurement study reveals that interference is typical and *partial* [14]. This means that transmissions from an interfering node may corrupt only a fraction of the packets received at a remote node. Modeling interference as contention results in a misleading optimization problem, where capacity may be wasted and actual receiving rates may be far from fair.

We have developed a new model of the wireless mesh network that accurately models partial interference. In this model, contention is still represented as an undirected edge between two vertices (links), however interference is modeled as a directional edge from the interfering link to the receiving link that is affected by the interference. Figure 2 shows a simple topology of a wireless mesh network, with the resulting contention graph shown in Figure 3. From this graph, we find the set of maximal cliques, which results in clique constraints. In our example of Figure 3, the constraints for Clique 1 and Clique 2, are the same as in the binary interference model, but the constraint between links 2 and 3 is modeled separately.

Interference relationships impose constraints on the receiving rates, as illustrated in the figure, where r_l is the effective receiving rate of link l, s_l is the sending rate of link l, d_l is the inherent loss of the link (e.g. due to obstacles or noise), c_j is the capacity of clique j, and the term $(1 - a_{il}s_i)$ is the loss of link l due to interference from an interfering node i. This constraint is taken from a recent measurement study of wireless mesh networks showing that partial interference from one link can be modeled as a linear function [14]. The interference factor a_{il} represents the degree of partial interference inflicted by the interferer. It is in the

range $0 \leq a_{il} \leq 1$, and is directional, meaning that a_{il} may be significantly different from a_{li}. Interfering factors can be experimentally measured between any pair of links in a network by methods suggested in [14, 16], constructing an interference map for the network. A study shows that interfering transmissions are independent of each other, and the joint impact of interferers to a receiving node is merely the product of their isolated impacts [14].

4.1.2 Problem Formulation

Given a contention graph with maximal cliques C and an interference map A, we formulate an optimization problem that maximizes the sum of link utilities, which are functions of link *receiving* rates, in a wireless mesh network:

$$\mathbf{P} : \max_{s} f(r) = \sum_{l \in L} U(r_l) \tag{1}$$

subject to:

$$s_l \geq 0, \ \forall l \in L, \tag{2}$$

$$r_l = d_l s_l \prod_{i \in I(l)} (1 - a_{il} s_i), \ \forall l \in L, \tag{3}$$

$$\sum_{l \in L(j)} s_l \leq c_j, \ \forall j \in C. \tag{4}$$

We assume the utility function U of a link is continuously differentiable, strictly concave, monotonically increasing, and approaches negative infinity as the argument approaches zero from the right.

Assuming a logarithmic utility function, and ignoring the ratio d_l because it does not affect the optimality of our solution, we can formulate this as a convex problem $\mathbf{P}'$

$$\mathbf{P}' : \max f'(s) \tag{5}$$

where

$$f'(s) = \sum_{l \in L} \left(\ln s_l + \sum_{i \in F(l)} \ln (1 - a_{li} s_l) \right). \tag{6}$$

subject to:

$$s_l \geq 0, \ \forall l \in L, \tag{7}$$

$$\sum_{l \in L(j)} s_l \leq c_j, \ \forall j \in C. \tag{8}$$

Note that a similar formulation that optimizes flow receiving rates, rather than link receiving rates, is not convex. Finding a convex reformulation is a part of our ongoing work.

4.1.3 Algorithm

Because the link formulation of this problem is convex, it is straightforward to derive a distributed algorithm to solve it, based on the methods presented in [13]. We use the gradient projection method to iteratively obtain the optimal price, λ, for the problem. Using a step size γ in the negative direction of the gradient gives the algorithm

$$\lambda_j(k+1) = \max\left(0, \lambda_j(k) - \gamma(c_j - \sum_{l \in L(j)} \bar{s}_l(\lambda(k)))\right), \tag{9}$$

where

$$\bar{s}_l(\lambda) = \arg\max_{s_l} g(s_l, \lambda). \tag{10}$$

and

$$g(s_l, \lambda) = \ln s_l + \sum_{i \in F(l)} \ln(1 - a_{li} s_l) - s_l \sum_{j \in C(l)} \lambda_j, \tag{11}$$

where $F(l)$ is the set of all links that interfere with l and $C(l)$ is the set of all links in a clique with l.

This means that each link in a maximal clique can share with each other their current rates, $\bar{s}_l$, which is only local information, and then compute the next price λ_j, which in turn leads each link to calculate new prices. The convergence of the algorithm is well established in the literature, even when it is asynchronous. Once λ converges to the optimal solution, the optimal solution, s^*, is given by

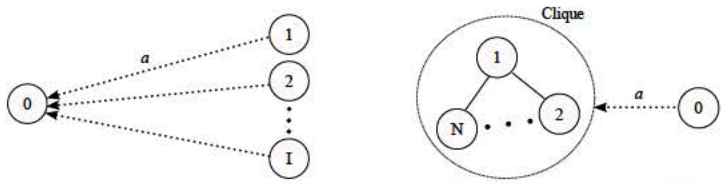
$$s^* = \bar{s}(\lambda^*).$$

4.1.4 Numerical Results

We use MATLAB to demonstrate the situations in which the partial interference (PI) model outperforms the binary interference (BI) model, and by how much. For binary interference, we assume that the model can either model interference as contention or ignore it, depending on which gives the largest utility. In most cases, the BI model ignores interference when it is low and models it as contention when it is high. To compare the models, we consider the ratio R of performance between the PI and BI models, using three topologies, each with clique capacities of 0.85.

Figure 4a shows the first topology, where I links interfere with a single link with a common interference factor a, but do not interfere with each other. Figure 5a shows the performance ratio for this topology, with the dotted curve showing when R begins to be greater than one. Interestingly, the PI model and the BI model perform exactly the same for values of a below 0.59. This is because, for low values of a, the cost of interference is offset by the gain of the interferer sending at full capacity. Thus, both the PI model and the BI model calculate sending rates at full capacity for each link. For larger values of a and I, the PI model outperforms BI more than 1.7 times.

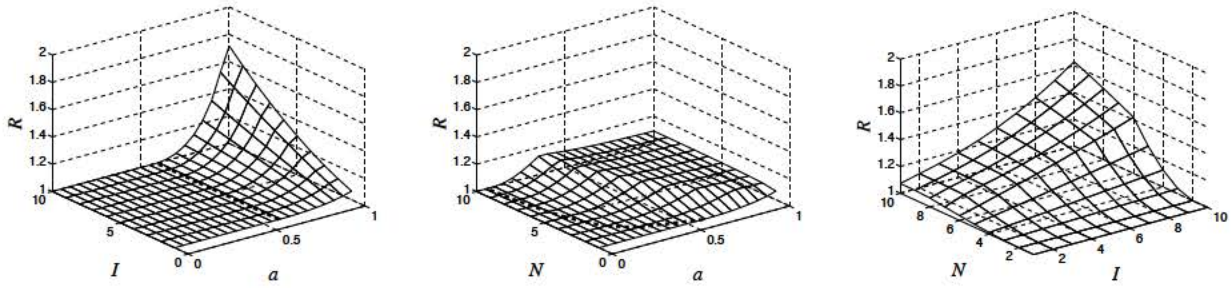
Figure 4b shows the second topology, where a single link has interference factor a on N links that contend in a single clique. Figure 5b plots the performance ratio for this topology, with the dotted curve showing where R begins to be greater than one. The PI model starts



(a) I links interfering with a single link.

(b) N contenders in a clique with one interferer.

Figure 4: Topologies used for numerical results.



(a) Ratio R of performance between the PI model and the BI model for I intererers on one link.

(b) Ratio R of performance between the PI model and the BI model, for N contenders with one interferer.

(c) Ratio R of performance between the PI model and the BI model, for I interferers and N contenders, with $a = 0.4$.

Figure 5: Numerical results for PI model.

performing better than the BI model at much lower values of a when N is large. This is due to the fact that the contending links already have small rates as a consequence of sharing the medium. Utilities are lowered much more by interference when sending rates are small. Thus, even for low values of a, the PI model does not calculate sending rates at full capacity. However, for higher values of a and N, the PI model outperforms the BI model only about 1.1 times.

To further demonstrate the worth of the PI model, we consider a topology combining features of the first two, where I links have a fixed interference factor $a = 0.4$ on N links that contend in a single clique. Figure 5c plots R for this topology. Experimental results show that it is typical for interference factors to range anywhere between zero and one in a real network, with usually at least one interferer on a link having a factor of at least $a = 0.8$, so choosing $a = 0.4$ in this topology is a somewhat conservative comparison [14]. The combined effect of several interferers and several contenders causes the PI model to perform significantly better than the BI model.

4.1.5 Implementation

To validate the performance benefits of the PI model, we implement a rate controller as an application for the Linux operating system, using the WiFu toolkit [4]. The primary challenges for the implementation are (1) constructing a network interference map, giving the contention and interference variables we use for our partial interference model, (2) based

on the interference map, forming a contention graph and enumerating maximal cliques, (3) implementing the rate controller algorithms, and (4) scheduling flows according to the calculated rates.

Interference Map To measure the network interference map, we use unicast transmissions between pairs of nodes, similar to the method discussed by Padhye et al. [15]. The measurement is performed in four steps:

1. Sending nodes select receivers. Each sending node selects a receiver to form two separate unicast links, say A→C and B→D. The receivers should be chosen in a way that transmissions from one sending node do not interfere with the packet reception of another link. This is to separate the impact of interference from contention.

2. Sending nodes take turns to transmit data to their respective receivers using unicast at full link capacity. Receivers calculate the transmission rate. In the example, C calculates R_a and D calculates R_b.

3. Sending nodes concurrently transmit data to their respective receivers using unicast at full link capacity. Receivers calculate the transmission rate. In the example, C calculates R_a^b and D calculates R_b^a. The node in the superscript is the potential contending node.

4. Receivers calculate the contention ratio. For node C, the contention ratio is defined as $\frac{R_a^b}{R_a}$, and D calculates the ratio as $\frac{R_b^a}{R_b}$. Note, the two contention ratios could be different for asymmetric contion. Node B contends with node A if $\frac{R_a^b}{R_a} < 1$, and similarly node A contends with B if $\frac{R_b^a}{R_b} < 1$.

The above steps are repeated for each permutation of nodes of interest for contention measurement. A pair of nodes is considered to be contending with each other if either one of the contention ratios is less than 1, because the partial interference model assumes symmetric contention between a pair of neighboring nodes.

The interference map is measured using a similar unicast approach. However, the interfering node should be within the interference range of the receiver and outside of the carrier sense range of the sender of the interferee link. To measure interference of node B to link A→C, node A first transmits to C at link capacity while B remains silent, and C calculates the receiving rate from A, R_a. In the next step, both A→C and B→D transmit at link capacity concurrently, and C calculates the receiving rate R_a^b when transmissions from B are present. Node B interferes with link A→C if $\frac{R_a^b}{R_a} < 1$, and the interference factor when B transmits at full link capacity is defined as $\left(1 - \frac{R_a^b}{R_a}\right)$.

Enumerating Maximal Cliques To calculate fair rates, links need to construct their local contention graphs and find the maximal cliques using the network interference and contention map. Unfortunately, enumerating maximal cliques in an arbitrary graph is a well-known NP-hard problem, and the problem is extremely difficult in dense graphs. With

the binary interference model, links within interference range of each other cannot be active concurrently. As a result, the contention graph is likely to be dense, because there are likely a large number of interferers to a remote node. Existing graph-based proposals adopt approximations by either over-conservatively assigning links within 2 hops apart to the same maximal clique [2], or requiring major modifications to underlying link layer protocols [9, 5].

With the partial interference model, it is viable to design efficient and accurate protocols that enumerate maximal the cliques in a contention graph. The work presented in [7] suggests that efficient algorithms exist for enumerating maximal cliques in graphs that are 1) sparse and 2) closed under the operation of taking subgraphs. The maximal cliques in the partial interference model only consist of links with senders within the carrier sense range of each other. In a typical wireless mesh network, the number of immediate contenders is quite limited, and we can assume the contention graph satisfies the above two assumptions.

Accordingly, we use the Bron-Kerbosch algorithm [3] to calculate all the maximal cliques that a link belongs to. The algorithm is executed within each node, and the CPU overhead is trivial as compared to the wireless communication overhead. Our method is able to work in real time as compared to previous work in the literature that conservatively assigns any node within two hops to the same clique.

Rate Controller Algorithms When implementing the rate controller algorithms, there are a number practical issues we must consider. First, the link-based formulation of the partial interference model doesn't capture what we really value – flow utility rather than link utility. To overcome this limitation, we set a weight on each link equal to the number of flows that traverse the link. Links with more flows will have heavier weights in the optimal rate control, and thus be assigned more bandwidth. To prevent unfairness between flows sharing the same link, each link equally divides the assigned bandwidth among its flows.

Second, the partial interference model is based on the assumption that each link has an infinite backlog of packets to send, which clearly may not hold in practice. This means that when a link is assigned a given fair rate, it may not actually be able to transmit packets at that rate. To deal with this problem, our implementation uses the actual transmission rates for each link when calculating prices, rather than the assigned rate. If a sending node transmits at a lower rate than expected due to insufficient packets, the sum rate of the clique becomes lower than the optimal target, and the clique price becomes lower, leading all links in the clique to increase their rates. In this way, flows with sufficient packets are able to utilize the idle channel. However, the rate allocation is unable to converge to the optimal target, because there is always a gap between the actual and the optimal transmission rate because of the insufficient packets.

Scheduling Flows Once the rate controller has determined the optimal rate for each link, we need to schedule the packets for each flow traversing each link, ensuring that the total rate of all flows on a link obey the rate limit. To do this, we use WiFu, which allows user-space programs on a Linux system to intercept, modify, and reschedule packets as they are forwarded. Figure 6 shows the architecture for our system.

WiFu intercepts packets using the Linux **iptables** software with the **netfilter** interface [1]. The intercepted packets are then given to a handler, depending on where the packet was

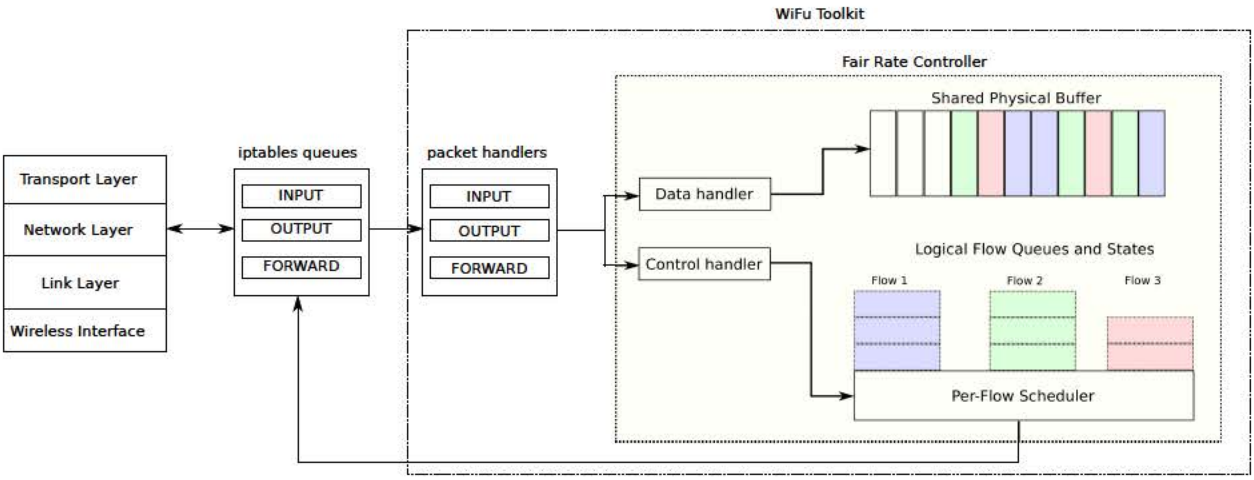


Figure 6: Fair rate control implementation architecture

intercepted, relative to the forwarding process. Control packets are send to a separate handler that the rate controller uses to calculate fair rates. These messages include notification from neighbors regarding their current link rates, which the rate controller uses to calculate a clique price. The rate controller uses a per-flow scheduler, with logical queues for each flow that traverses a link. Data packets are placed in a shared physical queue, and then transmitted by the scheduler according to their assigned rate.

4.1.6 Experiments

We evaluate the partial interference model on a wireless mesh testbed located in our department's building. One part of this testbed is shown in Figure 7. We run a number of experiments, but in this report focus a scenario where two connections are active: mesh25 $\rightarrow$ mesh 31 and mesh17 $\rightarrow$ mesh27. Due to the topology and signal strengths, the upper connection in this figure interferes with the lower connection, with an interference factor $a = 0.546$. This means that at full power the upper connection corrupts an average of 54.6% of the packets being sent on the lower connection.

As we might expect, the solution to the rate optimization problem in this case requires having the interferer slow down its sending rate to improve the performance of the interferee and maximize the sum of the link utilities. In this case, the upper connection should send at a normalized rate of 0.915 (1052 KBps) while the lower connection maintains a rate of 1.0 (1150 KBps). The receiving rate of the upper link should be 1052 Kbps while the lower rate should receive data at a rate of 575 Kbps. Figure 8a shows the actual received rates for each connection. Note that the lower connection receives only 475 KBps, lower than what is expected.

To investigate the reason for this discrepancy, we conducted an additional experiment where we vary the normalized rate of the interferer from 0 to 1, to determine if there is a better rate it could have chosen. Figure 8b shows the received rate for both connections as a function of the interferer's rate. This is a very surprising result, because the rate received by the lower connection is non-linear, contradicting results from the literature [14]. This pattern was observed with other pairs of nodes in our experiments, but we have not yet

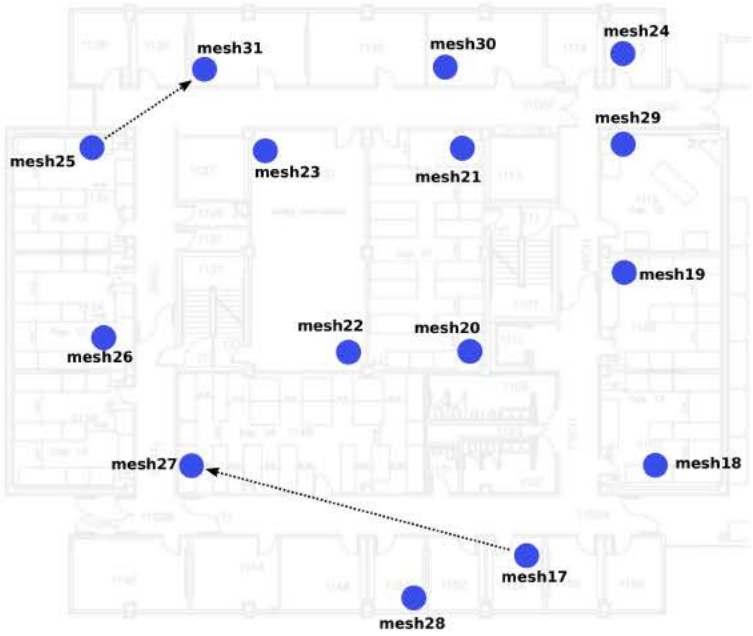
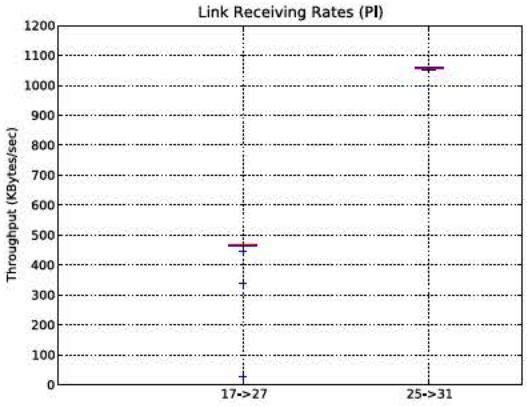
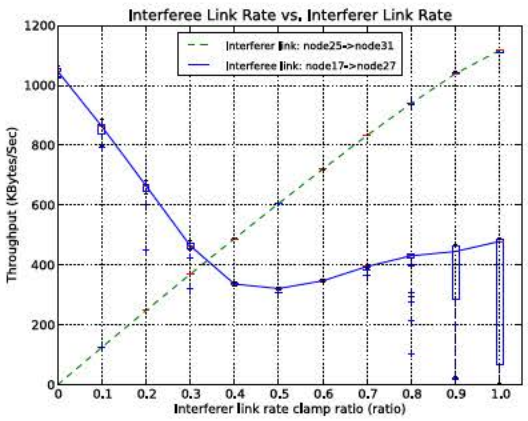


Figure 7: Wireless mesh topology



(a) Performance of PI model



(b) Non-linear relationship between interferer and interferee links

Figure 8: Experiments with partial interference model

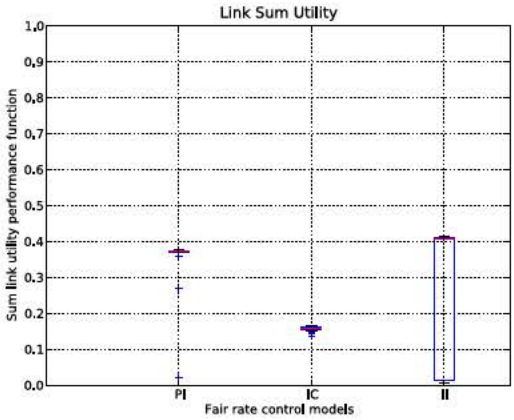


Figure 9: Sum utility

determined the exact reasons for it. This is a major area for future research, because this may require significant changes to our underlying model of the network.

Despite this finding, the partial interference model does perform better than the binary interference model on this topology, in terms of overall link utility. Figure 9 shows the link utility for the partial interference model as compared to binary interference, for (a) when interference is modeled as contention and (b) when interference is ignored. Modeling interference as contention is far too conservative, meaning it is better to let interfering links corrupt the packets of other links rather than making them take turns. Ignoring interference does very well, due to the non-linear behavior seen in Figure 8b. However, because the interferer is not rate limited at all, the lower connection suffers greatly and sometimes is completely starved. Thus even though the partial interference model is not exactly correct, there is some benefit to rate limiting.

4.2 First Principles Modeling of Wireless Networks

We further improve the accuracy of our wireless networks model by focusing on the constraints used to model the network, as this is the critical piece in the optimization approach. Here we use a first-principles model, meaning that we start with the most basic assumptions of how multi-hop wireless networks with CSMA operate. In this model, perceived times that the medium is occupied are represented as a random set. Our model may also be classified as a measurement based model, as it takes as inputs the probabilities of links carrier sensing or interfering with each other. Such an approach is more realistic than physical layer modeling, such as the various signal fading models with SINR thresholds, and has been used significantly to model wireless networks for various purposes [15, 8, 17, 16, 11, 14, 12].

4.2.1 Model

In this model, we make the following elementary assumptions:

- *Discretization of time.* Time is divided into large blocks of time that are further divided into equally sized slots. During each time slot, each link is either sends or doesn't.

- *Uniform random selection.* For each time block T, each link has a set $F \subset T$ of available time slots in which to send, and a set $X \subset F$ when it does send. Each $t \in F$ has an equal probability of being in X.
- *Negligible indirect scheduling.* Using the notation from the above assumption, if link i carrier senses links j and k sending during $X_j, X_k \subset T$, respectively, then dependencies of $X_j \cup X_k$ on the sending times of any link $l \neq i, j, k$ is negligible.

When considering the effective rate of links j and k as perceived by link i, realistically there may be some other link l (or even a set of other links) that cause the rates of j and k to overlap more or less than usual, which could in turn affect how much link i can send. The last assumption simply states that these effects are negligible. We recognize that it can significantly impact the accuracy of the model. This concern will be addressed in future research with empirical testing.

4.2.2 Problem Formulation

In this model, the sending constraint is given by

$$s_i + S_i \leq 1, \ \forall i \in L, \tag{12}$$

where

$$S_i = \sum_{p \in \mathcal{P}(L_i)} (-1)^{|p|-1} f_i(p) g_i(p) h(p), \tag{13}$$

$$f_i(p) = \prod_{j \in p} c_{ij} s_j, \tag{14}$$

$$g_i(p) = \frac{\phi_i(p)}{\prod_{j \in p} \phi_i(j)}, \tag{15}$$

$$\phi_i(p) = 1 - s_i \sum_{p' \in \mathcal{P}(p)} (-1)^{|p'|-1} \prod_{j \in p'} c_{ji}, \tag{16}$$

and the independence is given by

$$h(p) = \prod_{\{i,j\} \in \mathcal{P}_2(p)} (1 - c_{ij} - c_{ji} + c_{ij} c_{ji}). \tag{17}$$

The receiving constraint is given by

$$r_i = d_i (1 - R_i) s_i, \tag{18}$$

where

$$R_i = \sum_{p \in \mathcal{P}(L_i)} (-1)^{|p|-1} f_i'(p) h(p) \tag{19}$$

and

$$f_i'(p) = \prod_{j \in p} a_{ij} s_j. \tag{20}$$

In this model, $\mathcal{P}(p)$ is the set of all subsets of p except the empty set, and $\mathcal{P}_z(p)$ is the set of all subsets of p with $|p| = z$.

Note that (17) is an approximation. If one link carrier senses another link completely ($c_{ij} = 1$) then their random sets do not intersect. If any two random sets in p do not intersect, then the intersection of p is empty, which means that $h(p)$ should equal zero. Only when all random sets are independent should it equal one.

4.2.3 Numerical Results

Instances of this problem are frequently non-convex, due to the addition and subtraction of several rational functions in S_i, and due to similar reasons in R_i. We have developed a branch and bound solution to solve the problem by successively dividing the hypercube in which the feasible set resides into smaller regions, and evaluating lower and upper bound functions for the optimal value in each region. The bounds on each region allow one to conclude that some regions need not be divided further.

We use this solution to compare the performance of the first-principles model against the PI and BI models on several kinds of topologies. To do so, we define the optimality of a controller as

$$O = P'/P^*, \tag{21}$$

where P' is the performance (geometric mean of receiving rates) of the controller and P^* is the performance (lower bound) reported by the first-principles NUM problem. Note that O is simply the inverse of the performance ratio R used earlier. Thus an optimality of 1 equates to no performance loss despite the inaccuracies in the model with respect to partial carrier sensing or partial interference.

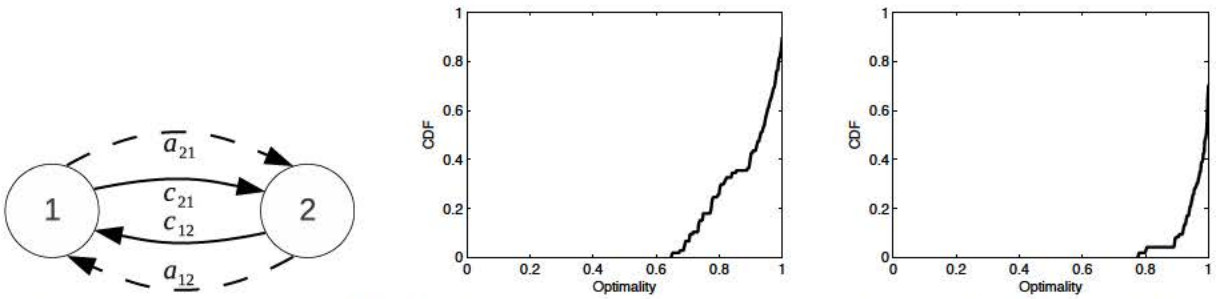
For the smaller topologies, the branch and bound algorithm converged to within a difference of 0.01 performance. However, for some of the larger topologies, the algorithm took too long to converge. We therefore will also report a certainty measure in these cases according to (21), where P' is the branch and bound's lower bound score and P^* is the upper bound score. This measure tells us how much higher the true optimal score might be, and thus how much worse the optimality of the controllers might be. If the certainty is not reported for a particular topology, then it was very close to 1.

The binary contention graphs for the partial and binary interference models were built based on the independence approximation (17). If two links had an independence greater than 0.5, an edge was drawn between them. For the maximal clique model, a contention edge also needed to be drawn for high interference values. This was done by defining a function mimicking the independence, but using a instead of c. An edge was drawn on the maximal clique model's contention graph if the multiplication of the two "independence" functions (a and c) was greater than 0.5.

When computing results over a range of interference factor values a, we omitted any topologies such that there existed two links i and j where

$$a_{ij} > 1 - c_{ij}. \tag{22}$$

This is because such topologies cannot occur due to the relationship between a and c. If two links carrier sense each other well, their sending rates cannot overlap much and therefore they cannot interfere with each other much.



(a) Contention/interference graph of the two-link topologies

(b) Optimality of the binary interference controller

(c) Optimality of the partial interference controller

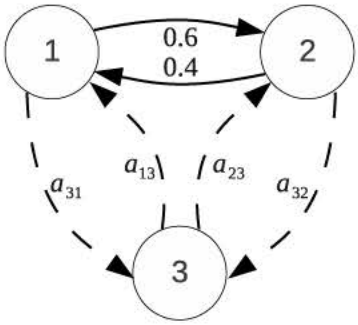
Figure 10: Optimality of the controllers for the two-link topologies

We first measured optimality on the simpliest topologies of interest, with only two links. The partial contention/interference graph of this class of topologies is given in Figure 10a. In this figure and subsequent figures, we let solid arrows denote contention (where the arrow points to the link that is sensing) and dashed arrows denote interference (where the arrow points to the link being interfered with). The values of a_{12}, a_{21}, c_{12}, and c_{21} were permutated over the range of 0 to 1, with granularity of 0.2, omitting the topologies that were unrealistic. The CDFs of the optimality for both models appear in Figure 10b and Figure 10c. This shows that the binary interference controller performs above 0.9 optimality in most cases, but many cases drop well below that. The partial interference controller, on the other hand, almost always performs above 0.9 optimality.

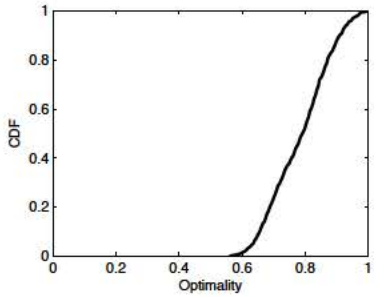
We next tested optimality over various three-link topologies by choosing the two-link topology with the worst optimality for the partial interference controller and adding a link to it. We first tested over all interference factors a both from and to the third link, with a granularity of 0.2, and c values involving link 3 set to zero. The partial contention/interference graph of this class of topologies is given in Figure 11a. The CDFs of the optimality for both models appear in Figure 11b and Figure 11c. For the binary interference controller, optimality is worst at 0.56, with various topologies scoring at or around this level. Optimality is close to 1 largely for the cases where there is high interference ($a \geq 0.8$) for most or all parameters. For the partial interference controller, 95% of all cases are at 0.85 optimality or better.

We also tested over all contention coefficients c both from and to the third link, with a granularity of 0.2, and a values involving link 3 set to zero. The partial contention/interference graph of this class of topologies is given in Figure 12a and the optimality CDFs in Figure 12b and Figure 12c. The two CDFs are identical because without any interference, the two classical models are identical. In general, these controllers lose performance when there is a significant amount of partial carrier sensing in the network (c values not close to 0 or 1). However, we have seen in the above results of varying the a parameters that introducing partial interference into a topology plagued with partial carrier sensing usually results in much better optimality for the partial interference controller.

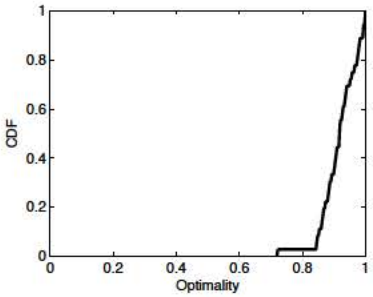
Another set of topologies we tested over was with I interferers on one link, with $a = 1$ and $c = 0.5$ between the interferers. Figure 13a shows the partial contention/interference



(a) Contention/interference graph of the three-link topologies varied over interference

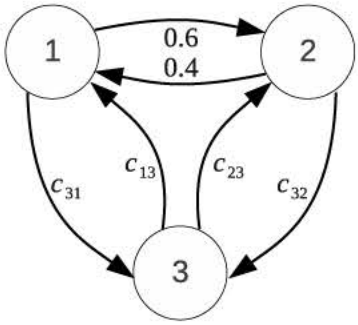


(b) Optimality of the binary interference controller

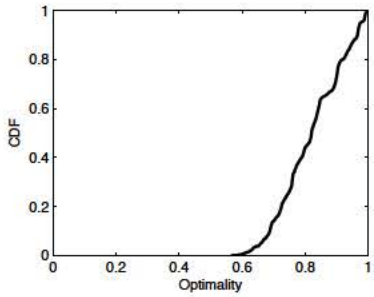


(c) Optimality of the partial interference controller

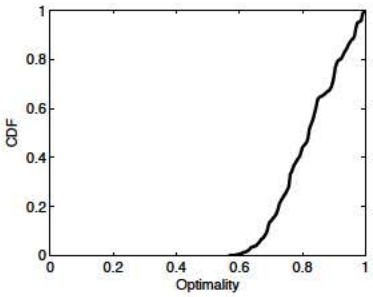
Figure 11: Optimality of the controllers for the three-link topologies varied over interference



(a) Contention/interference graph of the three-link topologies varied over contention

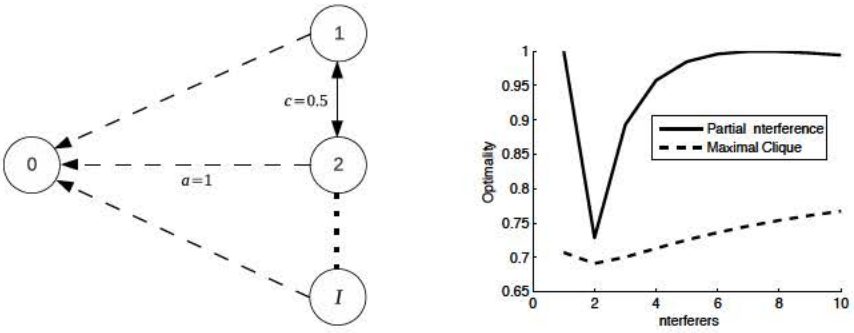


(b) Optimality of the binary interference controller



(c) Optimality of the partial interference controller

Figure 12: Optimality of the controllers for the three-link topologies varied over contention



(a) Contention/interference graph of the topologies with partially dependent interferers

(b) Optimality of binary and partial interference models for partially dependent interferers

Figure 13: Optimality of the controllers for partially dependent interferers

graph for this class of topologies. Figure 13b plots the optimality of the classical models across values of I. Only when $I = 2$ does the partial interference controller have significantly low optimality, whereas the maximal clique controller always does poorly. These results show that when contention between interferers is only partial, the partial interference controller performs much better in the face of complete interference ($a = 1$) than it does when interferers are completely dependent.

We next tested optimality on a chain topology of 5 nodes arranged in a line, with active links sending both ways between each node. Since many of the links are connected by a node, there is a significant amount of perfect contention ($c = 1$), typical of a real network. There is also interference among nodes that are farther away from each other. On this topology, the binary interference controller achieved an optimality of 0.861 and the partial interference controller achieved an optimality of 0.978. It is possible that the true optimalities are much worse, since the certainty of the first-principles branch and bound solution is only 0.709. However, based on the lengthy runs of the branch and bound algorithm we believe it is unlikely that the true optimal score is closer to the upper bound, since the lower bound never moves and the upper bound slowly approaches the lower bound. The branch and bound algorithm ran on this topology for approximately 16 hours and 60,000 iterations.

Finally, we tested optimality on a mesh topolgy, as shown Figure 14. We designed this topology to be similar to a typical flow of traffic through a mesh network providing Internet access to users, and is based on the positioning of nodes on one floor of a mesh network deployed in our department's building. It can be thought of as a multi-hop Internet access network for users, where nodes X and Y are the access points to the cloud. Note that just about any subset of the pairs of nodes could be the links that are simultaneously active, and the choice for this particular problem is somewhat arbitrary. The two parts of the network do contend and interfere with each other.

Like the chain topology, the mesh topology has a good mix of partial and binary contention, with some partial interference. The maximal clique controller obtained 0.924 optimality, the partial interference controller obtained 0.97 optimality, and the branch and bound algorithm returned a certainty of 0.807 for the optimal score, after running for several days and approximately 78,000 iterations. As a side note, a certainty of 0.795 was obtained

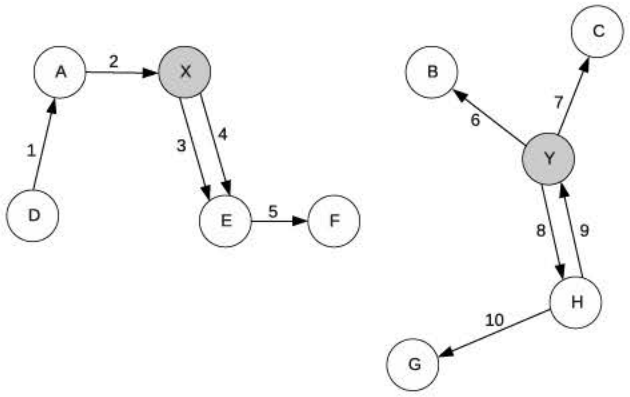


Figure 14: Network graph of the mesh topoogy

after only a few minutes, and the lower bound never moved, as was the case for all topologies tested.

The take-home message of these numerical results is that the maximal clique controller usually has significant performance loss, whereas the partial interference controller seems to only have significant performance loss in select topologies where the approximations introduced by the partial interference model sometimes cause the controller to either significantly under-utilize the medium or significantly starve at least links. Specifically, we have seen that when a network is plagued with a lot of partial carrier sensing, but also has very little interference, the partial interference model treat the partial carrier sensing as full carrier sensing, and thus under-utilize the medium. We have also seen that when a network has several interferers that can carrier sense each other (near) perfectly, *and* when that interference is very strong (nearly 1), the partial interference model will make the mistake of having the clique of interferers send at or close to the full clique capacity, thus starving the link being interfered with.

Despite these significant failures in specific instances for the partial interference model (optimalities as low as 0.33), the results on the larger networks of the chain and mesh topologies indicate that perhaps these dangerous motifs either do not occur frequently, or their detriment is washed out by the good performance in other parts of the network. In both topologies, the partial interference controller scored above 0.97. Of course, the results of the chain and mesh topologies cannot prove such high optimality for other topologies of similar size—but combined with the near-exhaustive search over the small topologies, it is a strong indication that the partial interference model is good enough for the purposes of rate control.

4.3 Robust dynamical network structure reconstruction

One of the fundamental interests in systems biology is the discovery of the specific biochemical mechanisms that explain the observed behaviour of a particular biological system. In particular, we consider the problem of reconstructing the network structure from input and partially measured output data of a dynamical system, and in turn uncovering the underlying mechanisms responsible for the observed behaviour. The biological network reconstruction

problem challenges come from the necessity to deal with noisy and partial measurements (in particular, the number of hidden/unobservable nodes and their position in the network is unknown) taken from a nonlinear and stochastic dynamical network.

There are several tools in the literature to infer causal network structures. These tools are mainly rooted in three fields: Bayesian inference [26, 40], information theory (ARACNe [21], [24], [27]) and ODE methods (inferelator [22], [19], [20], [28], [37]). Details on these and other methods can be found in several reviews of the field such as [18, 25, 30, 32, 36]. The vast majority of network reconstruction methods produce estimates of network structure regardless of the informativity of the underlying data. In particular, most methods produce estimates of network structure even in cases with data from only a few experiments. Such data may not contain enough information to enable the accurate reconstruction of the actual network, thus the obtained network estimates can be arbitrarily different from the true network structure [25]. To compensate for lack of information in data, most methods have heuristics that try to "guess" at the remaining information, either by specifying prior distributions or by appealing to a priori beliefs about the nature of real biological networks, such as looking for the sparsest network. Nevertheless, these heuristics bias the results and lead to incorrect estimates of the network structure.

In contrast, our approach has been to identify the conditions when data is sufficiently informative to enable accurate network reconstruction. The results indicate that even in an ideal situation, when the underlying network is linear and time-invariant (LTI) and the measurements are noise-free, network reconstruction is impossible without additional information [29]. Surprisingly, this information gap is not due to a lack of data, or a deficiency in the number of experiments, but rather it occurs because system states are only partially observed; the information gap is present in all data sets except those that satisfy certain experimental conditions. Our analysis identified a particular experimental protocol that satisfies these necessary conditions to ensure that data will be sufficiently informative to enable network reconstruction. This protocol suggests that:

1. A network composed of p measured species demands p experiments;

2. Each experiment requires a distinct input that independently controls a measured specie, i.e. experimental input i must affect measured specie i and no other measured specie except, possibly, indirectly through measured specie i.

If data acquisition experiments are not performed in this (or an equivalent) way, the network cannot be reconstructed. Moreover, the resulting information gap is catastrophic, meaning that any internal network structure explains the data equally well (i.e. fully decoupled, fully connected, and everything in between). On the other hand, if some information about the network is available a priori, as is usually the case, then these conditions can be relaxed as explained in [29].

The work in [29], however, did not take into account the realistic scenario that typically systems are nonlinear and data are noisy. This section extends and details earlier results in [44] by developing an effective method to reconstruct networks in the presence of noise and nonlinearities, assuming that the conditions for network reconstruction presented above in

(1) and (2) have been met. Steady-state (resp. time-series) data can be used to reconstruct the Boolean (resp. dynamical) network structure of the system.

The section is organised as follows. After a motivating example showing that input-output data alone does not enable network reconstruction, Section 7.3.1 reviews dynamical structure functions and gives fundamental results concerning their usefulness in the network reconstruction problem. Section 7.3.2 presents the main results of the section regarding robust network reconstruction from input-output data subject to noise and nonlinearities. Finally, we conclude this section with biologically-inspired examples in Section 7.3.3.

Notation. For a matrix $A \in \mathbb{C}^{M\times N}$, $A_{ij} \in \mathbb{C}$ denotes the element in the i^{th} row and j^{th} column while $A_j \in \mathbb{C}^{M\times 1}$ denotes its j^{th} column. For a column vector α, $\alpha[i]$ denotes its i^{th} element. We define $e_r^T = [0, \ldots, 0, 1_{r^{th}}, 0, \ldots, 0] \in \mathbb{R}^{1\times N}$. I denotes the identity matrix. When it is clear from the context, we omit the explicit dependence of transfer functions on the Laplace variable s, e.g. we write G instead of $G(s)$.

Motivating example. Consider the transfer function

$$G(s) = \frac{1}{s+3}\begin{bmatrix} \frac{1}{s+1} \\ \frac{1}{s+2} \end{bmatrix}$$

obtained from data (partial observations) using system identification tools. For simplicity, assume that $G(s)$ accurately represents the input-output relation of the original system. This transfer function is consistent with two state-space realisations $\dot{x} = Ax + Bu$, $y = Cx$ given by

$$A_1 = \begin{bmatrix} -1 & 0 & 1 \\ 0 & -2 & 1 \\ 0 & 0 & -3 \end{bmatrix}, \quad A_2 = \begin{bmatrix} -2 & -1 & 1 \\ -1 & -3 & 1 \\ 0 & -1 & -1 \end{bmatrix},$$

$B_1 = B_2 = [0 \quad 0 \quad 1]^T$, and $C_1 = C_2 = [I \quad 0] \in \mathbb{R}^{2\times 3}$ (i.e., the third state is hidden/non-observable). Note that both realisations are minimal and correspond to very different network structures as seen in Figure 15. This demonstrates that even in the idealised setting (LTI system, non noise and perfect system identification), network reconstruction in the presence of hidden/unobservable states is not possible without additional information about the system.

4.3.1 Dynamical structure functions and network reconstruction

In [29] we introduced the notion of dynamical structure functions and showed how they can be used to obtain necessary and sufficient conditions for network reconstruction. For the sake of clarity and completeness, we state these previously obtained results here without proofs. We refer the interested reader to [29, 41] for the corresponding proofs.

Consider a nonlinear system $\dot{\bar{x}} = f(\bar{x}, \bar{u}, w_1)$, $\bar{y} = h(\bar{x}, w_2)$ with p measured states $\bar{y}$, hidden states $\bar{z}$ (potentially a large number of them), m inputs $\bar{u}$, and noises w_1, w_2. The system is linearised around an equilibrium point (i.e., a point $(\bar{x}^*, \bar{u}^*)$ such that $f(\bar{x}^*, \bar{u}^*, 0) = 0$), and it is assumed that inputs and noises do not move the states too far from the equilibrium point so that the linearised system is a valid approximation of the original nonlinear system. The linearised system can be written as $\dot{x} = Ax + Bu$, $y = Cx$, where $x = \bar{x} - \bar{x}^*$, $u = \bar{u} - \bar{u}^*$ and $y = h(\bar{x}, 0) - h(\bar{x}^*, 0)$. The transfer function associated with this linearised system is

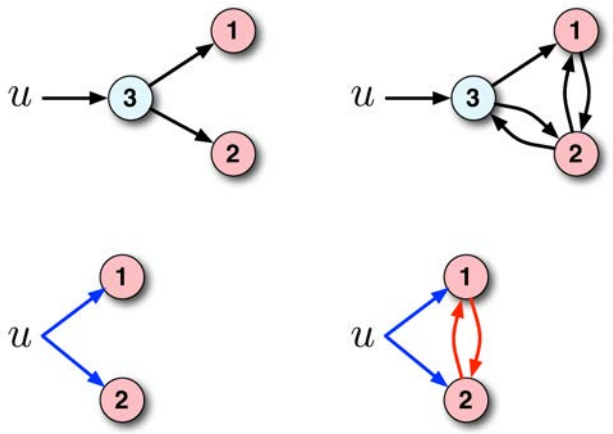


Figure 15: The same transfer function yields two minimal realisations with very different network structures (Left vs. Right). Pink nodes are measured, while blue nodes represent unmeasured hidden states; the top diagram on either side reveals the complete network structure explicitly showing hidden states, while the lower diagram indicates the corresponding casual structure captured by the dynamical structure function (edges associated with Q are red, while those associated with P are blue). The system in the left is (A_1, B_1, C_1) in 7.3, and the system on the right is (A_2, B_2, C_2). Note how completely different the two network structures are (complete decoupled vs. fully connected) even though either realization would be an equally valid description if all one knew about the system was its transfer function, identified from input-output data.

given by $G(s) = C(sI - A)^{-1}B$. When we have partial observations, i.e., when $C = [I \;\; 0]$, we partition the linearised system equation as follows

$$\begin{aligned} \begin{bmatrix} \dot{y} \\ \dot{z} \end{bmatrix} &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u \\ y &= \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} \end{aligned} \tag{23}$$

where $x = [y^T \; z^T]^T \in \mathbb{R}^n$, is the full state vector, $y \in \mathbb{R}^p$ is a partial measurement of the state (we assume $p > 1$), z are the $n - p$ "hidden" states, and $u \in \mathbb{R}^m$ is the control input. We restrict our attention to situations where output measurements constitute partial state information, i.e., $p < n$. Taking the Laplace transforms of the signals in (23), solving for Z, and substituting into the Laplace transform of the first equation of yields $sY = WY + VU$, where $W = A_{11} + A_{12}\left(sI - A_{22}\right)^{-1} A_{21}$ and $V = A_{12}\left(sI - A_{22}\right)^{-1} B_2 + B_1$. Now, letting D be the matrix composed of the diagonal elements of W, we write $\left(sI - D\right) Y = \left(W - D\right) Y + VU$. We then obtain $Y = QY + PU$ where

$$Q = \left(sI - D\right)^{-1}\left(W - D\right) \text{ and } P = \left(sI - D\right)^{-1} V. \tag{24}$$

Given the system (23), we define the *dynamical structure function* of the system to be (Q, P). If all the measured states are removed from the system except for Y_i and Y_j then the transfer function Q_{ij} corresponds to the exact transfer function between Y_j (considered as input) and Y_i (considered as output). The same holds for P in terms of U_j and Y_i.

It can be shown that $G = (I - Q)^{-1} P$ (see [29]). Based on this latter relation, it can be seen that the dynamical structure function of a system contains more information than the transfer function, and less information than the state-space representation [29]. We can then conclude that, with no other information about the system, neither dynamical nor Boolean reconstruction is possible. Moreover, for *any* internal structure Q there is a dynamical structure function (Q, P) that is consistent with G, i.e., that satisfies $G = (I - Q)^{-1} P$. In particular, this shows that the use of criteria such as sparsity or decoupledness to guide our selection of a proposal network structure can be misleading. If one were to optimise for decoupledness, for example, a dynamical structure $(0, G)$ could and would always be found, regardless of the true underlying structure. Thus, if we are to use these kinds of criteria, they must be firmly justified a priori.

Proposition 1. *[29] Given a $p \times m$ transfer function G, dynamical structure reconstruction is possible from partial structure information if and only if $p - 1$ elements in each column of $(Q, P)^T$ are known that uniquely specify the component of (Q, P) in the nullspace of $[\ G^T \quad I\ ]$.*

The importance of this result is that it identifies exactly what information about a system's structure, beyond knowledge of its transfer function, must be obtained to be able to recover the structure without appeal to a priori assumptions, such as sparsity, or parsimony, etc. This enables the design of experiments targeting precisely the additional information needed for reconstruction. In particular when $p = m$ and G is full rank, we observe that imposing that P is diagonal, i.e., that each input controls a measured state independently, is sufficient for reconstruction.

Corollary 1. *[29] If $m = p$, G is full rank, and there is no* a priori *information about the internal structure of the system, Q, then the dynamical structure can be reconstructed if each input controls a measured state independently, i.e., if, without loss of generality, the inputs can be numbered such that P is diagonal.*

4.3.2 Robust network structure reconstruction

In this section, we consider the problem of robustly reconstructing dynamical network structures. Data are obtained from input-output measurements of a noisy nonlinear system. From this type of data we aim to find the internal network structure Q associated with the linearised system (23). To average out the noise, data-collection experiments are repeated N times. For simplicity of exposition, we assume that no *a priori* information on the internal network structure Q is available. The results still follow if some *a priori* information about Q is available, and such information can typically be used to relax the experimental protocol according to Proposition 1. Hence, data are collected according to the measurement protocol described in the introduction:
(1) the number of distinct data-collection experiments is the same as the number of measured species. This in particular implies that $u(t), y(t) \in \mathbb{R}^p$;
(2) each input u_i controls first the measured state y_i so that P is a $p \times p$ diagonal matrix.

In the following two subsections (7.3.2 and 7.3.2), we propose two approaches for estimating the dynamical structure function (Q, P) from measured input-output data. The first

approach is indirect and involves estimating the transfer function G followed by computing (Q, P) from G. Since some information is lost in the process of estimating G, we consider a second approach where (Q, P) is directly estimated from data (without estimating first G). Concerning the type of input-output data collected, we first consider time-series input-output data and then the special case where only steady-state data are available.

Dynamical network reconstruction from identified transfer functions This section describes a method to obtain the dynamical structure function from a transfer function G. This transfer function was identified from noisy time-series data using standard system identification tools [33]. According to Corollary 1, if G is full rank there is a unique Q and diagonal P satisfying $(I - Q)G = P$. Since G is an approximation of the actual system, Q and P will typically be mere approximations of the actual dynamical structure function. Moreover, due to noise and unmodelled dynamics, it is likely that Q does not even have the correct Boolean structure. Typically, the internal structure function Q obtained from such a procedure will be fully connected, i.e., all non-diagonal elements of Q will be non-zero.

The main idea to solve the network reconstruction problem from noisy data is the following. For p measured states, Q has $p^2 - p$ unknowns. We want to quantify the smallest *distance* from G (or directly from the measured data) to all possible Boolean structures (and there are 2^{p^2-p} of them). Some of such distances will be large revealing that the corresponding Boolean structures are unlikely to be the correct structures while other will be small making them candidates for the correct structure.

There are a number of ways to model input-output data with noise and nonlinearities. In order to obtain a convex minimisation problem, we consider the output (could also be input) feedback uncertainty model [43]. In this framework, the "true" system is given by $(I+\Delta)^{-1}G$, where Δ represents unmodelled dynamics, including nonlinearities, and noise. Based on this choice of dynamic uncertainty, the distance from data to a particular Boolean structure is chosen to be $\|\Delta\|$, in some norm, such that Q obtained from $(I + \Delta)^{-1}G = (I - Q)^{-1}P$ has the desired Boolean structure. We can rewrite the above equation as $\Delta = GP^{-1}(I - Q) - I$. Now, let $X = P^{-1}(I - Q)$. Then the Boolean structure constraint on Q can be reformulated on X, i.e., non-diagonal zero elements in X correspond to those in Q (since $X_{ij} = P_{ii}^{-1}Q_{ij}$ for $i \neq j$).

We can order all Boolean structures from 1 to 2^{p^2-p}, and define a set $\mathcal{X}_k$ containing transfer matrices that satisfy the following conditions: (i) for $i \neq j$, $X_{ij}(s) = 0$ if for the considered k^{th} Boolean structure $Q_{ij}(s) = 0$; all other $X_{ij}(s)$ are free variables; (ii) when $i = j$, $X_{ii}(s)$ is a free variable. Hence, the distance from G to a particular Boolean structure can be written as $\alpha_k = \inf_{X \in \mathcal{X}_k} \|GX - I\|^2$, which is a convex minimisation problem with a careful choice of a norm. Next, we show that this problem can be cast as a least squares optimisation problem. If we use the norm defined by $\|\Delta\|^2 = \text{sum of all } \|\Delta_{ij}\|_2^2$, where $\|\cdot\|_2$ stands as the $\mathcal{L}_2$-norm over $s = j\omega$, then using the projection theorem [39] the problem

reduces to

$$\begin{aligned}\alpha_k = \inf_{X\in\mathcal{X}_k} \|GX - I\|^2 &= \inf_{X\in\mathcal{X}_k} \sum_i \|GX_i - e_i\|_2^2 \\ &= \sum_i \inf_{Y_i} \|A_i Y_i - e_i\|_2^2 \\ &= \sum_i \|A_i(A_i^*A_i)^{-1}A_i^* e_i - e_i\|_2^2,\end{aligned}$$

where X_i is the i^{th} column of $X \in \mathcal{X}_k$, Y_i is a column vector composed of the free (i.e., nonzero) elements of X_i, A_i is obtained by deleting the j^{th} columns of G when the corresponding elements $X_i[j]$ are 0 for all j, and $(\cdot)^*$ denotes transpose conjugate. The infimum is achieved by choosing $Y_i = (A_i^*A_i)^{-1}A_i^*e_i$, and $A_i^*A_i$ is always invertible since G is full rank in Corollary 1. If experiments are repeated N times, yielding a transfer function G^i for each experiment, then the above analysis still follows simply by letting $G = \left[G^{1^T} \ \cdots \ G^{N^T}\right]^T$.

Dynamical network reconstruction directly from time-series data The previous sections used a two-step approach in which system identification was first used to estimate a transfer function from measured input-output data and then, in a second step, the identified transfer function was used to obtain a dynamical structure function representation of the system which is optimal in terms of a particular metric. This section proposes a method which allows identification of the optimal dynamical structure function representation directly from the measured input output data. The advantage of this direct network structure reconstruction from data is that no information is lost during the initial transfer function identification stage.

Due to the equivalence between dynamical uncertainty perturbations [43], we are free to chose, without loss of generality, the type of uncertainty perturbation that best suits our needs. For the direct method, instead of a feedback uncertainty as was considered in the previous section, the uncertainty perturbation we are considering here is the additive dynamic uncertainty on the output, i.e., $Y = G_\Delta(U + \Delta)$. In this case, we think about the "distance" in terms of how much we need to change the input (data) to fit a particular Boolean structure. Since $G_\Delta = (I - Q)^{-1}P = X^{-1}$, the equality $Y = G_\Delta(U + \Delta)$ can be written as

$$\Delta = XY - U,$$

where $X \in \mathcal{X}_k$, for some particular Boolean network k. Recall that structural constraints in Q can be imposed directly on X from the equality $X = P^{-1}(I - Q)$. We can therefore use system identification tools for non-causal autoregression models under the structural constraints to identify X (which might be non-causal). In this case, the distance is defined as the maximum likelihood of the estimation problem.

Penalising connections The above methodology suffers from a crucial weakness: there are several Boolean structures with distances smaller or equal than the distance to the "true" network. Indeed, the extra degrees of freedom of the fully-connected network allow its corresponding distance α_k to be the smallest of all. This is similar to the noisy data

over-fitting problem encountered in system identification where the higher the order of the transfer function, the better the fit. The typical approach in system identification is to penalise higher dimensions and the analogy here is to penalise extra network connections.

If the true network has l non-existent connections (l off-diagonal elements in Q are zero) then there are $2^l - 1$ different Boolean networks that have a smaller or equal distance (due to the additional degrees of freedom provided by the extra connections). When noise is present, then the "true" network will typically have an optimal distance similar to these other l networks. The question of how to find the "true" network thus arises. With repeated experiments, small enough noise (i.e., large enough signal-to-noise ratio) and negligible non-linearities, the optimal distances of those l networks are comparable, and they are typically much smaller than those of the other networks. To try to reveal the "true" network, one can strike a compromise between network complexity (in terms of number of connections) and data fitness by penalising extra connections. There are several ways to do this. Here, we consider one of the classical methods known as Akaike's information criterion (AIC) [31], or some of its variants such as AICc (which is AIC with a second order correction for small sample sizes), and the Bayesian information criterion (BIC) [23].

The AIC-type approach is a test between models - a tool for model selection. Given a data set, several competing models may be ranked according to their AIC value, with the one having the lowest AIC being the best. From the AIC value one may typically infer that the best models are in a tie and the rest are far worse, but it would be arbitrary to assign a threshold above which a given model is rejected [23]. The AIC value for a particular Boolean network B_k is defined as:

$$AIC_k = 2L_k - \ln \alpha_k, \tag{25}$$

where L_k is the number of (non-zero) connections in the Boolean network B_k and α_k is the optimal distance based on this parameter constraint.

Although finding the optimal distance in the second term of eq. (25) can be done efficiently, the number of Boolean networks 2^{p^2-p} grows very fast with the number of measured states p. To find the network with the smallest distance it is thus not desirable to compute the optimal distance for each possible Boolean network. Fortunately, there are ways to reduce the number of networks that need to be considered. As we saw in the previous section $\inf_{X\in\mathcal{X}_k} \|GX - I\|^2 = \sum_i \inf_{Y_i} \|A_i Y_i - e_i\|_2^2$ meaning that we can solve each optimisation problem separately. Since each Y_i corresponds to $p-1$ unknowns in the i^{th} row of Q, this reduces the problem to solving $p2^{p-1}$ optimal distances. Finding a polynomial-time algorithm to compute the optimal distance through this method is a subject of current investigation. When it comes to the steady-state case, [19] proposed a polynomial-time algorithm to quickly find the ranked solutions at the expense of solution accuracy.

Boolean network reconstruction from steady-state data So far we have assumed that time-series data are available. Frequently, however, experimentation costs and limited resources only permit steady-state measurements. In addition, with steady-state measurements it is typically possible to perform a larger number of experiments within the same amount of time, effort and cost. As shown below, most of the connectivity of the network together with the associated steady-state gains (and the associated positive or negative sign) can still be reconstructed from steady-state data. However, no dynamical information will

be obtainable. In other words, for most cases we can still recover the Boolean network from steady-state data.

Assume that after some time of maintaining the control input concentrations at a constant value, the measured outputs y have converged to a steady-state value. This is equivalent (if the system is stable or quasi-stable [37]) to assuming that we can obtain $G(0)$, i.e., $G(s)$ evaluated at $s = 0$. Now the relationship $(I - Q(s))G(s) = P(s)$ evaluated at $s = 0$ becomes $(I - Q(0))G(0) = P(0)$. From this equation and the knowledge of $G(0)$, all of the results given in Sections 7.3.2 and 7.3.2 follow provided that no element of $G(s)$ has a system zero [43] at 0. In that case, a nonzero element in the obtained Boolean network indicates the existence of a causal relationship between the corresponding pair of nodes while a zero element indicates the absence of such relationship.

4.3.3 Biologically-inspired examples

This section illustrates with two examples the theoretical results presented in the previous section. The corresponding sets of ordinary differential equation describing the dynamics of the considered networks are used to generate noisy data, which are then fed to our reconstruction algorithm in order to assess its ability to recover the correct network structure.

Single feedback loop In this first example, we consider the following nonlinear system:

$$\dot{y}_1 = -y_1 + \frac{V_{max}}{K_m + z_3^3} + u_1 \tag{26}$$

$$\dot{y}_2 = -2y_2 + 1.5z_1 + u_2 \tag{27}$$

$$\dot{y}_3 = -1.5y_3 + 0.5z_2 + u_3 \tag{28}$$

$$\dot{z}_1 = 0.8y_1 - 0.5z_1 \tag{29}$$

$$\dot{z}_2 = 1.2y_2 - 0.8z_2 \tag{30}$$

$$\dot{z}_3 = 1.1y_3 - 1.3z_3 \tag{31}$$

where $V_{max} = 0.5$ and $K_m = 0.1$. Equation (26) includes a nonlinear function of z_3 known as a Hill equation. It represents a negative regulation of the rate of reaction of y_1 by z_3. For simplicity, all other terms are linear. In this example, $p = 3$, i.e., there are three measured states (y_1, y_2 and y_3) while the other 3 states are hidden (z_1, z_2 and z_3). The corresponding network is given in Figure 16(a).

Three experiments were performed. In each experiment, one input was a step while the others were set to zero and data was collected for each of the measured species. The experiments were repeated 3 times to average out the noise. For simplification, in this example, only steady-state data was used. Data was obtained by numerically integrating the differential equations in (26)-(31) and adding independent Gaussian noises. The ratios between standard deviations and means of the steady-state data were within the range $[0.35, 1.15]$, which shows that noise is considerable.

Since the true network has 3 elements in Q equal to zero, there are $2^3 = 8$ networks with a better or equal optimal cost. Computing the corresponding distances and AICc values for all possible Boolean structures between the three measured species, we observed that

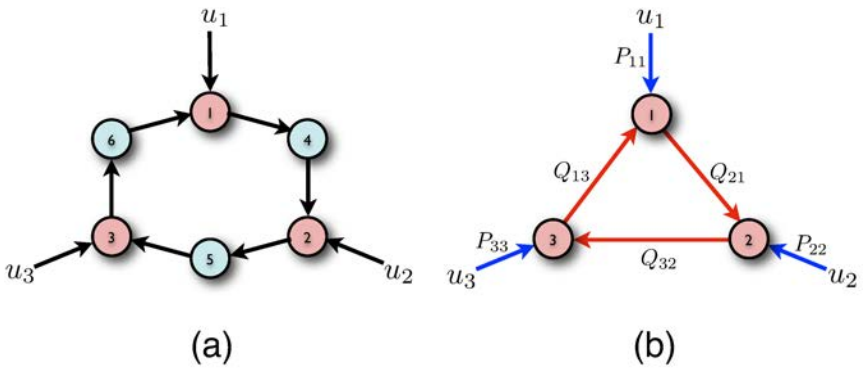


Figure 16: (a) Complete network with all the states. The red circles represent the measured states while the blue circles correspond to hidden states. (b) Network of the measured states only.

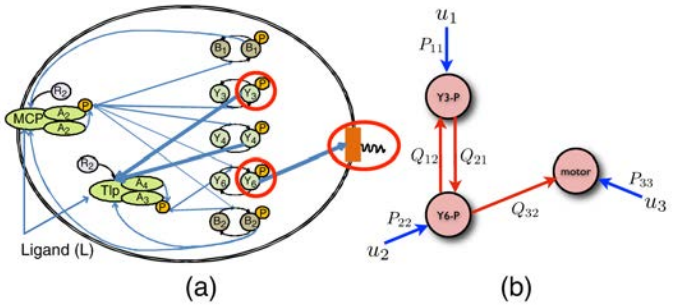


Figure 17: (a) Network representing the dynamical interaction between the 10 species believed to be responsible for the chemotactic response of *Rhodobacter sphaeroides*. We assume that only species Y_3^p, Y_6^p and "motor" are measured (circled in red). (b) Network connecting the measured states only.

the distance decreases by an order of magnitude when we arrived at the true network. In addition, AIC, BIC and in particular AICc are able to pick the correct network.

Chemotaxis in *Rhodobacter sphaeroides* This section considers the reconstruction of the biochemical network responsible for chemotaxis in *Rhodobacter sphaeroides*. The network is represented in Figure 17 (see [35, 38] for a detailed explanation of this model and its biological interpretation). It involves 10 species dynamically interacting through a complex set of interconnections. To illustrate our method, consider noisy data from 3 species only: Y_3^p, Y_6^p and the "motor" (circled in red in Figure 17(a)), obtained based on simulations of the nonlinear ordinary differential equation model proposed by [35]. We follow our prescribed experimental protocol and, for simplification, only steady-state data are used. Relatively large Gaussian noise was added to the collected data to simulate measurement noise in the data set.

Based on the complete network given in Figure 17(a), the correct network to recover is presented in Figure 17(b). Computing the corresponding distances and AICc values for all the $2^6 = 64$ possible Boolean networks, we observed that the network with the smallest

AICc was not the correct network in Figure 17(b) as it was missing the Q_{12} link. A closer look at the noisy steady-state data of Y_3^p (from a step input in u_2) revealed an extremely large ratio between its standard deviation and mean value (≈ 200), showing that the noise was completely overpowering the signal. Indeed, it can be shown that Y_6^p has a very small influence on Y_3^p since the pathway from Y_6^p to Y_3^p includes a reversible reaction with a very small rate constant (for detail, see [42, 34]). The next set of smallest values of AICc consists of 4 networks, including the true one. If necessary, an extra experiment can be performed to further discriminate between these five candidate networks.

4.3.4 Discussion

This section proposes a new network reconstruction method in the presence of noise and nonlinearities based on dynamical structure functions. The key idea is to find minimal distances between the existent data and the data required to obtain particular network structures. The method was illustrated with two biologically-oriented examples. They showed that even in the presence of nonlinearities and considerable noise network reconstruction was possible. Eventually, when the signal to noise ratio was too small, reconstruction was no longer possible, but that is true irrespective of the method used.

Obviously, the method has limitations with respect to nonlinearities. With stronger nonlinear terms eventually the method fails. For example, network reconstruction for oscillatory systems is still an open problem. However, when applied to the reconstruction of various equilibrium point models given in the literature, we observed that reconstruction was always possible when the signal-to-noise ratio of the measured data was not too small (far less than 1).

A final note regarding the application of this methodology to real data. We have looked throughout the literature for real data and none of the available data that we found satisfied the conditions necessary for accurate network reconstruction. Some problems that we observe in the literature include: 1) many publications do not include raw data (they typically only include means and standard deviations), and many authors indicate that they no longer have their data; 2) some microarray data do not include repeats and others were obtained using dual channel microarrays that only give ratios between channels, making it impossible to reliably extract gene expression intensities. One of the most promising papers was [25], which followed our experimental protocol, and their raw data is available. We found, however, that the data did not meet the conditions necessary for network reconstruction. In the paper, the authors compared different network reconstruction methods only to find that none of the methods even came close to identifying the true network. Nevertheless, because the authors compared the results to random guessing, they report that "Reverse engineering based on differential equations and Bayesian networks correctly inferred regulatory interactions from experimental data." We disagree with their conclusion, and point out that because the data was not sufficiently informative in the first place, such a comparison is not meaningful. To better understand the degree of the lack of information in the data, we considered all 10 subnetworks consisting of 3 nodes. The gap in distances between fully decoupled and fully connected networks ranged from just 2% to a maximum of 70%. This shows that there is not enough information in the data to differentiate between Boolean structures. Note that this data was obtained from over-expression, so based on these results, we hypothesise that over-

expressing genes saturates the translation and transcription machinery, making linearisation a poor approximation of the actual system dynamics. Current work is exploring the design of experiments on known systems that 1) satisfy our data collection protocol to ensure the resulting data is sufficiently informative for network reconstruction, and 2) facilitate a comparison of various methods so we can better understand how different techniques perform in situations where accurate network reconstruction is, in fact, possible.

4.4 Minimal realization of dynamical structure functions and its application to network reconstruction

Recently, networks have received an increasing amount of attention. In our information-rich world, the questions of network reconstruction and network analysis become crucial for the understanding of complex systems such as biological, social, or economical networks. In particular, the analysis of molecular networks has gained significant interest due to the recent explosion of publicly available high-throughput biological data. In this context, the question of identifying and analyzing the network structure at the origin of measured data becomes a key issue.

In some occasions, measured data is given in the form of input-output time-series that describes the effect of inputs on outputs (measured states) of a network. When data is generated by a linear system, a matrix transfer function describing the dynamic input-output behavior is generally obtained using system identification [33]. If the original state-space model is available or deducible, then the associated network structure can be readily obtained from it. However, a transfer function cannot, in general, recover, or realize, the original state-space model since the realization problem does not typically have a unique solution, i.e., different state-space realizations can generate the same input-output behavior. Since each of these realizations may suggest entirely different network structures, it is in general impossible to identify network structures from transfer functions alone. Therefore, more information, beyond input-output data used to identify a transfer function, is needed to prefer one state-space realization over another as a description of a particular system[45].

Another difficulty in the network reconstruction problem comes from the fact that the realization problem becomes ill posed when some of the states are unobservable or "hidden" (this even happens with just one hidden state [43, pp. 78]). As a result, failure to explicitly acknowledge the presence of hidden states and the resulting ambiguity in network structures can lead to a deceptive and erroneous process for network structure discovery. Consequently, determining from measured data the presence or absence of a causal relationship between two variables in a network is a challenging question.

Motivated by this, we are focusing on the effect of hidden states in the network that we are aiming to reconstruct. A new representation for LTI systems, called dynamical structure functions was introduced in [29]. Dynamical structure functions capture information at an intermediate level between transfer function and state space representation (see Figure 18). Specifically, dynamical structure functions not only encode structural information at the measurement level, but also contain some information about hidden states. Based on the theoretical results presented in [29], we proposed some guidelines for the design of an experimental data-acquisition protocol which allows the collection of data containing sufficient

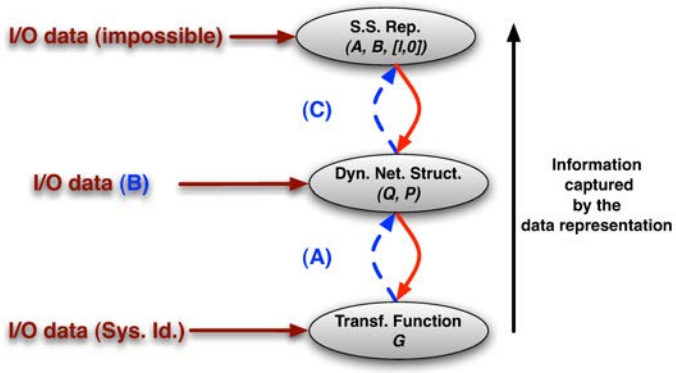


Figure 18: Mathematical structure of the network reconstruction problem using dynamical structure functions. Red arrows mean "uniquely determine", blue arrows indicate our work.

information for the network structure reconstruction problem to become solvable. In particular, we have shown that if nothing is known about the network, then the data-collection experiments must be performed as follows:

(A.1) for a network composed of p measured species, the same number of experiments p must be performed;

(A.2) each experiment must independently control a measured species, i.e., control input i must first affect measured species i.

If the experiments are not performed in this way the network cannot be reconstructed, and any network structure fits the data equally well (e.g. a fully decoupled network or a fully connected network). If biologists have already some information about the network, as it is usually the case, then these conditions can be relaxed as explained in [29].

Using dynamical structure functions as a mean to solve the network reconstruction problem, the following aspects need to be considered (see Figure 18):

First (see (A) in Figure 18), the properties of a dynamical structure function and its relationship with the transfer function associated with the same system need to be precisely established (this was done in [29]).

Second (see (B) in Figure 18), an efficient method is developed to reconstruct networks in the presence of noise and nonlinearities (this was done in [46]). This method relies on the assumption that the conditions for network reconstruction presented above in (A.1) and (A.2) have been met. In our approach, we use the same information as traditional system identification methods, i.e., input-output data. However, with our method, steady-state (resp. time-series data) can be used to reconstruct the Boolean (resp. dynamical network) structure of the system (see [46] for more details).

Third (see (C) in Figure 18), once the dynamical structure function is obtained, as a main result of this section, an algorithm for constructing a minimal order state-space representation consistent with such function is developed. In an application, this provides a way to estimate the complexity of the system by determining the minimal number of hidden states in the system. For example, in the context of biology it helps understand the number of unmeasured molecules in a particular pathway: a low number means that most

molecules in that pathway have been identified and measured, showing a good understanding of the system; while a large number shows that there are still many unmeasured variables, suggesting that new experiments should be carried out to better characterize that pathway.

The outline of the section is as follows. Section 7.4.1 reviews the definition of dynamical structure functions and their properties. The main result can be found in Section 7.4.2 where we propose a minimal order realization algorithm based on state-space realizations and pole-zero analysis. Simulation and discussion are addressed in Section 7.4.3. Finally conclusions are presented in Section 7.4.4.

4.4.1 System Model

Consider a linear system (it can also be a linearization of some original nonlinear system) $\dot{x} = Ax + Bu$, $y = Cx$. The transfer function associated with this system is given by $G(s) = C(sI - A)^{-1}B$. Typically, we can use standard system identification tools [33] to identify a transfer function $G(s)$ from input-output data.

Like system realization, network reconstruction also begins with the identification of a transfer function, but it additionally attempts to determine the network structure between measured states without imposing any additional structure on the hidden states. As we have shown in [29], this requires a new representation of linear time-invariant systems: the dynamical structure function (defined later). An algorithm allowing the dynamical structure function to be obtained from input-output data is proposed in [46]. This section assumes that the dynamical structure function has already been obtained from data, and will focus on finding one of its minimal state-space realizations.

The dynamical structure function is obtained as follows: First, we transform $[A, B, C]$ to $\left[A^o, B^o, \left[I_p \quad 0\right]\right]$ without changing $G(s)$, where $p = \text{rank}(C)$. The linear system dynamics then writes

$$\begin{aligned} \begin{bmatrix} \dot{y} \\ \dot{z} \end{bmatrix} &= \begin{bmatrix} A^o_{11} & A^o_{12} \\ A^o_{21} & A^o_{22} \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} + \begin{bmatrix} B^o_1 \\ B^o_2 \end{bmatrix} u \\ y &= \begin{bmatrix} I_p & 0 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix} \end{aligned} \tag{32}$$

where $x = (y, z) \in \mathbb{R}^{n^o}$ is the full state vector, $y \in \mathbb{R}^p$ is a partial measurement of the state, z are the $n^o - p$ "hidden" states, and $u \in \mathbb{R}^m$ is the control input. In this work we restrict our attention to situations where output measurements constitute partial state information, i.e., $p < n^o$. We consider only systems with full rank transfer functions that do not have entire rows or columns of zeros, since such "disconnected" systems are somewhat pathological and only serve to complicate the exposition without fundamentally altering our conclusions.

Taking the Laplace transforms of the signals in (32) yields

$$\begin{bmatrix} sY \\ sZ \end{bmatrix} = \begin{bmatrix} A^o_{11} & A^o_{12} \\ A^o_{21} & A^o_{22} \end{bmatrix} \begin{bmatrix} Y \\ Z \end{bmatrix} + \begin{bmatrix} B^o_1 \\ B^o_2 \end{bmatrix} U \tag{33}$$

where Y, Z, and U are the Laplace transforms of y, z, and u, respectively. Solving for Z gives

$$Z = (sI - A^o_{22})^{-1} A^o_{21} Y + (sI - A^o_{22})^{-1} B^o_2 U$$

Substituting this last expression of Z into (33) then yields

$$sY = W^o Y + V^o U \tag{34}$$

where $W^o = A^o_{11} + A^o_{12}\left(sI - A^o_{22}\right)^{-1} A^o_{21}$ and $V^o = B^o_1 + A^o_{12}\left(sI - A^o_{22}\right)^{-1} B^o_2$.

Now, let R^o be a diagonal matrix formed of the diagonal terms of W^o on its diagonal, i.e., $R^o = \text{diag}\{W^o\} = \text{diag}(W^o_{11}, W^o_{22}, ..., W^o_{pp})$. Subtracting $R^o Y$ from both sides of (34), we obtain:

$$\left(sI - R^o\right) Y = \left(W^o - R^o\right) Y + V^o U$$

Note that $W^o - R^o$ is a matrix with zeros on its diagonal. We thus have:

$$Y = QY + PU \tag{35}$$

where

$$Q = \left(sI - R^o\right)^{-1}\left(W^o - R^o\right) \tag{36}$$

and

$$P = \left(sI - R^o\right)^{-1} V^o \tag{37}$$

Note that Q is zero on the diagonal.

Definition 1. *Given the system (32), we define the dynamical structure function of the system to be* $[Q, P]$.

Note that, in general, $Q(s)$ and $P(s)$ carry a lot more information than $G(s)$. This can be seen from the equality $G(s) = (I - Q(s))^{-1}P(s)$ (see [29] for details). However, $Q(s)$ and $P(s)$ carry less information than the state-space model (32) (see [46]).

Definition 2. *A dynamical structure function,* $[Q, P]$*, is said to be consistent with a particular transfer function,* G*, if there exists a realization of* G*, of some order, and of the form (32), such that* $[Q, P]$ *are specified by (36) and (37). Likewise, a realization is consistent with* $[Q, P]$ *if that realization gives* $[Q, P]$ *from (36) and (37).*

Definition 3. *We say that a realization is* G *minimal if this realization corresponds to a minimal realization of* G*. We say that a realization is* $[Q, P]$ *minimal if this realization is consistent with* $[Q, P]$ *and its order is smaller than or equal to that of all realizations consistent with* $[Q, P]$.

The underlying principle to find a $[Q, P]$ minimal realization is to search for a realization with the minimal number of hidden states. Such a realization is characterized by the minimal number of pole-zero cancellations in the transfer functions Q and P.

Proposition 2. *Given a dynamical system (32) and the associated dynamical structure functions* $[Q, P]$ *with* R^o *constructed as explained above (see* (32)-(37)*), the following conditions must hold*

$$diag\{A^o_{11}\} = \lim_{s\to\infty} R^o(s); \tag{38}$$

$$A^o_{11} - diag\{A^o_{11}\} = \lim_{s\to\infty} sQ(s); \tag{39}$$

$$B^o_1 = \lim_{s\to\infty} sP(s). \tag{40}$$

Proof. Eq. (38) is directly obtained from the definition of $R^o(s)$:

$$\begin{aligned}\lim_{s\to\infty} R^o(s) &= \lim_{s\to\infty} \operatorname{diag}\{W^o(s)\} \\ &= \operatorname{diag}\{\lim_{s\to\infty} W^o(s)\} = \operatorname{diag}\{A^o_{11}\}\end{aligned}$$

Since the proofs for eq. (39) and (40) are very similar, we focus on eq. (39) only. Using the fact that for any square matrix M, $(I-M)^{-1} = \sum_{i=0}^{\infty} M^i$, we obtain, from the definition of Q given in (36), $Q(s) = \sum_{i=1}^{\infty} s^{-i} {R^o}^{i-1}(W^o - R^o)$ and $W^o = A^o_{11} + \sum_{i=1}^{\infty} s^{-i} A^o_{12} {A^o_{22}}^{i-1} A^o_{21}$. Hence, $Q(s) = (A^o_{11} - R^o(s))s^{-1} + r(s)$, in which $r(s)$ is a matrix polynomial, whose largest degree is -2. Finally, multiplying by s on both sides and taking the limit as s goes to ∞ results in eq. (39). A similar argument can be used to prove eq. (40). □

We give an illustrative example here.

Example 1. *Consider a system with the structure depicted in Fig. 19. A linear system's representation is*

$$\begin{aligned}\dot{x} &= \begin{bmatrix} a_{11} & 0 & a_{13} & 0 & 0 \\ 0 & a_{22} & 0 & a_{24} & 0 \\ 0 & a_{32} & a_{33} & 0 & a_{35} \\ a_{41} & 0 & 0 & a_{44} & 0 \\ 0 & a_{52} & 0 & 0 & a_{55} \end{bmatrix} x + \begin{bmatrix} b_{11} & 0 \\ 0 & b_{22} \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} u \\ y &= \begin{bmatrix} I_3 & 0 \end{bmatrix} x\end{aligned}$$

where I_3 is the 3×3 identity matrix. Following the definitions in (36) *and* (37)*, we can write down the corresponding dynamical structure function $[Q, P]$ with*

$$Q = \begin{pmatrix} 0 & 0 & \frac{a_{13}}{s-a_{11}} \\ \frac{a_{24}a_{41}}{(s-a_{22})(s-a_{44})} & 0 & 0 \\ 0 & \frac{a_{35}a_{52}+a_{32}(s-a_{55})}{(s-a_{33})(s-a_{55})} & 0 \end{pmatrix},$$

$$P = \begin{pmatrix} \frac{b_{11}}{s-a_{11}} & 0 \\ 0 & \frac{b_{22}}{s-a_{22}} \\ 0 & 0 \end{pmatrix}.$$

To illustrate Proposition 2, we have

$$\lim_{s\to\infty} sQ(s) = \begin{pmatrix} 0 & 0 & a_{13} \\ 0 & 0 & 0 \\ 0 & a_{32} & 0 \end{pmatrix};$$

$$\lim_{s\to\infty} sP(s) = s \begin{pmatrix} \frac{b_{11}}{s-a_{11}} & 0 \\ 0 & \frac{b_{22}}{s-a_{22}} \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} b_{11} & 0 \\ 0 & b_{22} \\ 0 & 0 \end{pmatrix}.$$

Generally, there exist many realizations consistent with $[Q, P]$. In the following section, we focus on finding a $[Q, P]$ minimal realization $(A, B, \begin{bmatrix} I & 0 \end{bmatrix})$, i.e., a realization which is consistent with $[Q, P]$ and which has minimal order, i.e., with the dimension of A minimal (and hence the lowest possible complexity).

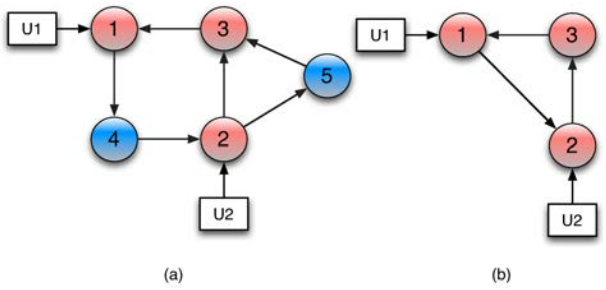


Figure 19: **(a)** An example system with two inputs, three measured states (states 1, 2, and 3) and two hidden states (states 4 and 5). **(b)** The corresponding dynamical network structure.

4.4.2 Algorithm to find a $[Q, P]$ minimal realization

From a dynamical structure function $[Q, P]$ we cannot reconstruct $[W^o, V^o]$ since there is no information regarding the diagonal transfer function matrix R^o. Given $[Q, P]$ and a diagonal proper transfer function matrix R, a minimal realization of $[W \;\; V] = [(sI - R)Q + R \;\; (sI - R)P]$ can be obtained as follows:

$$[W \;\; V] = [A_{11} \;\; B_1] + A_{12}(sI - A_{22})^{-1}[A_{21} \;\; B_2] \tag{41}$$

The idea is to start with an arbitrarily chosen R, and then use a state-space realization approach to find a R^* which minimizes the order of a minimal realization of $[W \;\; V]$.

Lemma 1. *Suppose W, A_{22} and A are defined as in eq. (41), then V and G share the same zeros.*

Proof. Since $sI - W$ is the Schur complement of $sI - A_{22}$ in $sI - A$, then

$$\det(sI - W) = \frac{\det(sI - A)}{\det(sI - A_{22})}. \tag{42}$$

Recall that $V = B_1 + A_{12}(sI - A_{22})^{-1}B_2$. Since $(sI - W)G = V$, we thus have that V and G share the same zeros [49, page 153]. □

Given a dynamical structure function $[Q, P]$, a random choice of a proper diagonal transfer function matrix R is likely to result in additional zeros in $V = (sI - R)P$. From Lemma 1, this will lead to additional zeros in G which are associated to uncontrollable eigenvalues of the considered realization [47, Section 4] and of course does not lead to a minimal realization in eq. (41). At this stage the following question arises: how can we find a proper diagonal transfer function matrix R^* such that a minimal realization of $[W \;\; V]$ is a $[Q, P]$ minimal realization, i.e.,

$$R^* = \mathbf{argmin}_R \deg\left\{(sI - R)s^{-1}[sQ \;\; sP] + [R \;\; 0]\right\}, \tag{43}$$

where deg is the McMillan degree [43]. Note that, since there are many choices for R^* that minimize the order of minimal realizations of $[W \;\; V]$, a chosen R^* may be different from R^o.

Assume that all elements in $[Q \;\; P]$ only have simple poles. This assumption can be relaxed but we adopt it here for simplicity. Also assume that $[Q \;\; P]$ does not possess any poles at 0 (otherwise we can change eq. (43) to $(sI-R)(s-a)^{-1}[(s-a)Q \;\; (s-a)P]+[R \;\; 0]$, where $a \in \mathbb{R}$ is not a pole of $[Q \;\; P]$).

Proposition 3. *Assume $[I-Q \;\; P]$ only has simple poles and does not have any zeros. A minimal order realization of $[W \;\; V]$ in (41) can be achieved using a constant diagonal matrix R^*.*

Proof. Assume R^* has at least one term on the diagonal with the degree of numerator greater or equal to 1, e.g., suppose the i^{th} term in $(sI-R^*)s^{-1} = \frac{(s+b)\epsilon_i(s)}{s\phi_i(s)}$ with any $b \in \mathbb{R}$ and $deg(\epsilon_i(s)) = deg(\phi_i(s)) \geq 1$, where $deg(\cdot)$ returns the degree of a polynomial. Hence, the multiplication $(sI-R^*)s^{-1}[sQ \;\; sP]$ will introduce $deg(\phi_i(s))$ new poles and, due the assumption of simple poles, can at most eliminate $deg(\epsilon_i(s)) = deg(\phi_i(s))$ poles. As a consequence, we can change the i^{th} term to $\frac{s+a}{s}$ without increasing the order. Doing this along all the elements of R^* proves the result. □

If R^* is a constant matrix, the term $[R^* \;\; 0]$ in eq. (43) is also a constant matrix. Therefore, the order of a minimal realization is only determined by $(sI-R^*)s^{-1}[sQ \;\; sP] \triangleq N[sQ \;\; sP]$. Thus, finding the "optimal" R^* which leads to the minimal order in eq. (43) is equivalent to finding a diagonal proper transfer matrix N (N with corresponding minimal realization (A_2, B_2, C_2, I) is restricted to the set of matrices of the form $(sI-R^*)s^{-1}$ with a constant R^* from Proposition 3) such that $N[sQ \;\; sP]$ has as few poles as possible. Based on this idea, the following algorithm is proposed:

Step 1: *Find a Gilbert's realization of the dynamical structure function.*

First, using the results in [29, Lemma 1], we find a minimal realization (A_1, B_1, C_1, D_1) of $[sQ \;\; sP]$. When $[sQ \;\; sP]$ has l simple poles, using Gilbert's realization [48] gives

$$[sQ \;\; sP] = \sum_{i=1}^{l} \frac{K_i}{s-\lambda_i} + \lim_{s\to\infty}[sQ \;\; sP],$$

where $K_i = \lim_{s\to\lambda_i}(s-\lambda_i)[sQ \;\; sP]$ and has rank 1 since we are assuming that $[sQ \;\; sP]$ has simple poles.

Consider a matrix decomposition of K_i in the following form:

$$K_i = E_i F_i, \;\; \forall i,$$

where $E_i \in \mathbb{R}^p$ and $F_i = (E_i^T E_i)^{-1} E_i^T K_i$. Then $A_1 = \text{diag}\{\lambda_i\} \in \mathbb{R}^{l\times l}$,
$B_1 = \begin{bmatrix} F_1^T & F_2^T & \dots & F_l^T \end{bmatrix}^T$, $C_1 = \begin{bmatrix} E_1 & E_2 & \dots & E_l \end{bmatrix}$ and $D_1 = \lim_{s\to\infty}[sQ \;\; sP]$.

Step 2: *Find the maximal number of cancelled poles.*

We define Φ as a largest subset of $\{E_1, \cdots, E_l\}$ such that all the elements in Φ are mutually orthogonal. We also define ϕ as the cardinality of Φ. Computationally, ϕ can be obtained using the algorithm presented in the Appendix. We claim that ϕ is equal to the maximum number of poles we can eliminate (the proof is in the Appendix). Therefore, the minimal order of $[W \;\; V]$ is

$$l - \phi.$$

As a consequence, the order of the minimal reconstruction is the dimension of A_{11} (constant p) plus the minimal dimension of A_{22} (obtained above): $p + l - \phi$.

Step 3: *Construct R^* to obtain the minimal reconstruction.*

Once we have Φ, we know that $N(\lambda_i)[j,j] = 0$ implies $R^*[j,j] = \lambda_i$. Consequently, each element in the set Φ will determine at least one element in R^*. This last fact can be used to construct R^* element by element. Once R^* is found, we can obtain $A,\ B$ using eq. (41).

4.4.3 Illustrative example

Example 2. *Consider a dynamical structure function $[Q, P]$:*

$$[Q \mid P] = \left[\begin{array}{ccc|c} 0 & \frac{1}{s+2} & \frac{1}{s+3} & \frac{1}{s+4} \\ \frac{1}{s+1} & 0 & \frac{1}{s+3} & \frac{1}{s+4} \\ \frac{1}{s+1} & \frac{1}{s+2} & 0 & \frac{1}{s+4} \end{array}\right].$$

We first compute the McMillan degree of the corresponding transfer function: $deg\{G\} = deg\{(I-Q)^{-1}P)\} = 4$, meaning that a 4^{th} order state-space model is enough to realize the transfer function. It is interesting to see what is the minimal order realization consistent with the dynamical structure function. The different steps of the algorithm proposed in the previous section successively yield the following:

Step 1: *A minimal Gilbert realization of $s[Q,P]$ is*

$$A_1 = diag\{-1,-2,-3,-4\},\ B_1 = diag\{2,2,2,4\},$$
$$C_1 = \begin{bmatrix} 0 & -1 & -1.5 & -1 \\ -0.5 & 0 & -1.5 & -1 \\ -0.5 & -1 & 0 & -1 \end{bmatrix},\ D_1 = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}.$$

Step 2: *By definition, $E_i = C_1 v_i$ where $v_i \in \mathbb{R}^4$ has 1 in its i^{th} position and zero otherwise. Thus,*

$$\{E_1, \cdots, E_4\} = \left\{ \begin{bmatrix} 0 \\ -0.5 \\ -0.5 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} -1.5 \\ -1.5 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \right\}.$$

Furthermore, ϕ is 1 and the order of a minimal realization of the given dynamical structure function is $p+l-\phi = 3+4-1 = 6$. Hence, the system must contain at least 3 hidden states.

Step 3: *R^* can be chosen as $diag\{a,-1,-1\}$, $diag\{-2,a,-2\}$, $diag\{-3,-3,a\}$, or $diag\{-4,-4,-4\}$ for any $a \in \mathbb{R}$.*

The reconstructed networks are represented in Fig. 20. There are three measured (red) nodes, labeled $1,2,3$ and by the analysis above, there are at least three hidden nodes such that the corresponding realization is consistent with the dynamical structure function. The red connections between measured nodes are the same for all candidate networks which is in accordance with Proposition 2. Dashed lines correspond to the connections between hidden and measured nodes.

From a biological perspective, this indicates that there are at least 3 unmeasured species interacting with the measured species. Of course, the "true" biological system might be even more complicated, i.e., it might have more than 6 species. Yet, when more states are measured, the dynamical structure functions can be easily updated and a new search for a minimal

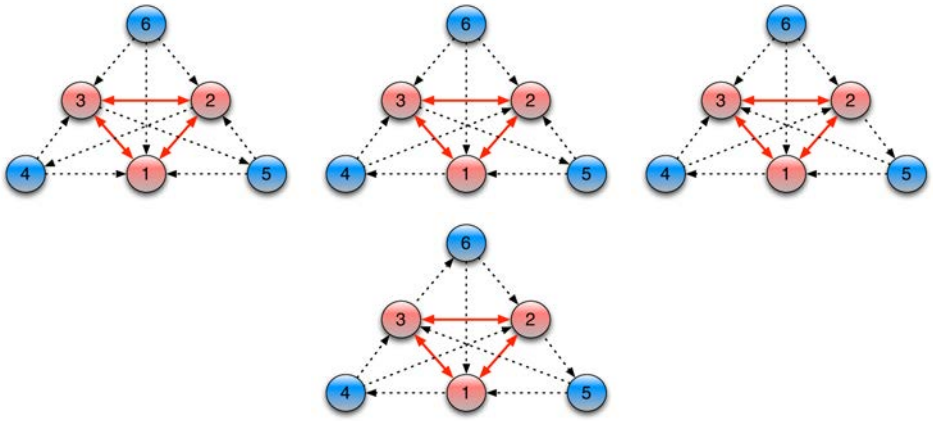


Figure 20: Topologies corresponding to the four $[Q, P]$ minimal realizations. The measured nodes are colored red, while the hidden ones blue. Red connections between measured nodes are the same for all the networks due to Proposition 2. Each node has a self-loop but we omit it for simplicity.

realization of the updated system can be performed to reveal the corresponding minimal number of hidden states.

4.4.4 Summary

In this section, we have presented a method for obtaining a minimal order realization consistent with a given dynamical structure function. We show that the minimal order realization of a given dynamical structure function can be achieved by choosing a constant diagonal matrix R^*. This provides a way to estimate the complexity of the system by determining the minimal number of hidden states that needs to be considered in the reconstructed network. For example, in the context of reconstruction of biological networks from data, it helps to understand the minimal number of unmeasured molecules in a particular pathway.

4.5 The Meaning of Structure in Interconnected Dynamic Systems

Structure and dynamic behavior are two of the most fundamental concepts characterizing a system. The interplay between these concepts has been a central theme in control as far back as Black's, Bode's, or Nyquist's work on feedback amplifiers [53, 54, 55], or even as early as Maxwell's analysis *On Governors* in 1868 [56].

The key property driving such analyses is the fact that an interconnection of systems yields another system. This property suggests a natural notion of structure, as the interconnection of systems, and focuses attention on understanding how interconnections of different systems result in varieties of dynamic behaviors.

This idea of structure as interconnection is not only critical in the analysis of systems, but it also plays a key role in system modeling. While black box approaches to modeling seek to duplicate the input-output behavior of a system irrespective of structure, first principles

approaches to modeling use the visible interconnection structure of a system to decompose the system into smaller, fundamental components. Constitutive relations describing each of these components are then applied, and interconnection principles linking these relations then result in a model structurally and dynamically consistent with the original system.

While such approaches have demonstrated remarkable success in electrical and mechanical domains where system components are readily visible and physically separated from each other [57, 58, 59], application of such methods to biological, social, and other domains has been more difficult. One reason may be that these systems do not exhibit a natural structure in the same sense that previous applications have; while components of electrical and mechanical systems are compartmentalized and solid-state, the physical relationship among components of these other systems are often much more fluid [60, 61]. Perhaps for these other domains different notions of structure play the role historically occupied by the interconnection of components.

This section explores these ideas by characterizing the complete computational structure of a system and then contrasting it with three distinct partial structure representations. These different representations include the interconnection of subsystems and the standard idea of a transfer function matrix, but it also includes a newer concept of system structure called *signal structure* that appears to be especially useful for characterizing systems that are difficult to compartmentalize. Precise relationships between these various perspectives of system structure are then provided, along with a brief discussion on their implications for various questions about realization and approximation.

4.5.1 Complete Computational Structure

The complete computational structure of a system characterizes the actual processes it uses to sense properties of its environment, represent and store variables internally, and affect change externally. At the core of these processes are information retrieval issues such as the encoding, storage, and decoding of quantities that drive the system's dynamics. Different mechanisms for handling these quantities result in different system structures.

Mathematically, state equations, or their generalization as descriptor systems [62, 63, 64] are typically used to describe these mechanisms. Although there may be many realizations that describe the same input-output properties of a particular system, its complete computational structure is the architecture of the *particular realization* fundamentally used to store state variables in memory and transform system inputs to the corresponding outputs. In this work we will focus our attention on a class of differential algebraic systems that are equivalent to a set of ordinary differential equations in state space form; we will refer to such equations as *generalized state equations*.

Representing a system's complete computational structure is thus a question of graphically representing the structure implied by the equations that govern its state evolution. In this work, rather than focusing on the specific syntax of any one particular graphical modeling language, we will draw from the standard system theoretic notions of a *block diagram* and a *signal flow graph* to conduct a concrete analysis between graphical representations of a system at various levels of abstraction. The complete computational structure of a system, then, is the description of the system with the most refined resolution, which we will characterize as a graph derived from a particular block diagram of the generalized state

equations.

To make this concept of structure precise, we begin by considering a system G with generalized state space realization

$$\begin{aligned} \dot{x} &= f(x, w, u), \\ w &= g(x, w, u), \\ y &= h(x, w, u). \end{aligned} \tag{44}$$

Note that this system is in the form of a differential algebraic equation, although we will only consider systems with differentiation index zero, implying that (44) is always equivalent to a standard ordinary differential or difference equation of the same order [65]. Typically we may consider the system (44) to be defined over continuous time, with $t \in \mathbb{R} \geq 0$, and with $u \in \mathbb{R}^m$, $x \in \mathbb{R}^n$, $w \in \mathbb{R}^l$, $y \in \mathbb{R}^p$, and $\dot{x}$ taken to mean dx/dt. Moreover, we restrict our attention to those functions f, g and h where solutions exist for $t \geq 0$. Nevertheless, we could also consider discrete time systems, with $t = 0, 1, 2, 3, ...$ and $\dot{x}$ in (44) taken to mean $x[t+1]$, or systems with general input, state, auxiliary, and output spaces $\mathcal{U}$, $\mathcal{X}$, $\mathcal{W}$, or $\mathcal{Y}$, respectively. In some situations these "spaces" may merely be sets, e.g. $\mathcal{X} = \{0, 1\}$. In any case, however, we will take $u \in \mathcal{U}^m$, $x \in \mathcal{X}^n$, $w \in \mathcal{W}^l$, and $y \in \mathcal{Y}^p$ so that m, n, l and p characterize the dimensions of the input, state, auxiliary and output vectors, respectively.

Note that the auxiliary variables, w, are used to characterize intermediate computation in the composition of functions. Thus, for example, we distinguish between $f(x) = x$ and $f(x) = 2(.5x)$ by computing the latter as $f(w) = 2w$ and $w = g(x) = .5x$. In this way, the auxiliary variables serve to identify stages in the computation of the state space realization (44). Frequently we may not require any auxiliary variables in our description of the system; indeed it is the standard practice to eliminate auxiliary variables to simplify the state descriptions of systems. Nevertheless, as we discuss structure, it will be critical to use auxiliary variables to distinguish between systems with dynamically equivalent, yet structurally distinct architectures, leading to the following definition.

Definition 4. *Given a system (44), we call the number of auxiliary variables, l, the* intricacy *of the realization.*

To understand the structure of (44), we need a notion of dependence of a function on its arguments. For example, the function $f(x, y, z) = xy - x + z$ clearly depends on z, but it only depends on x when $y \neq 1$ (or on y when $x \neq 0$). Since "structure" refers at some level to the dependence of the system variables on each other, it is important that our notion of dependence be made clear.

Definition 5. *A function $f(w)$, from l-dimensional domain $\mathcal{W}$ to s-dimensional co-domain $\mathcal{Z}$, is said to* depend *on the i^{th} variable, w_i, if there exist values of the other $s-1$ variables w_j, $j \neq i$, such that $f(w)$ is not constant over all values of w_i while holding the others variables fixed. If $s = 1$, then $f(w)$* depends *on w if it is not constant over all values of w.*

Note that when $\partial f/\partial w_i$ is well defined, the above definition coincides with the partial derivative being non-zero for some value of the variables w_j. Nevertheless, here we allow for non-differentiable functions as we explicitly characterize one notion of the structure of a state space realization.

Definition 6. *Given a system G with realization (44), its* complete *or* computational structure *is a weighted directed graph $\mathcal{C}$ with vertex set $V(\mathcal{C})$, and edge set $E(\mathcal{C})$. The vertex set contains $m+n+l+p$ elements, one associated with the mechanism that produces each input, state, auxiliary, and output variable of the system, and we label the vertices accordingly. In particular, the vertex associated with the i^{th} input is labeled u_i, $1 \leq i \leq m$, the vertex associated with the j^{th} state is labeled f_j, $0 \leq j \leq n$, the vertex associated with the j^{th} auxiliary variable is labeled g_j, $0 \leq j \leq l$, and the vertex associated with the k^{th} output is labeled h_k, $1 \leq k \leq p$. The edge set contains an edge from node i to node j if the function associated with the label of node j depends on the variable produced by node i. Moreover, the edge (i, j) is then labeled (weighted) with the variable produced by node i.*

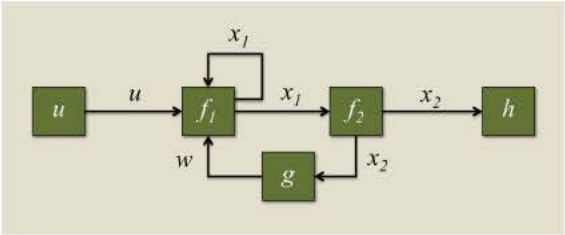
So, for example, consider the following continuous time system with real-valued variables:

$$\begin{array}{rcl} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} & = & \begin{array}{l} f_1(x_1, w, u) \\ f_2(x_1) \end{array} \\ w & = & g(x_2) \\ y & = & h(x_2) \ . \end{array} , \tag{45}$$

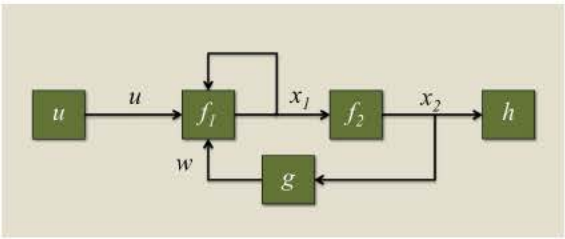
Its complete computational structure has node set $V(\mathcal{C}) = \{u, f_1, f_2, g, h\}$ and edge set $E(\mathcal{C}) = \{u(u, f_1), x_1(f_1, f_1), x_1(f_1, f_2), x_2(f_2, g), x_2(f_2, h), w(g, f_1)\}$; Figure 21 illustrates the graph.

This notion of the complete computational structure of the system (44) corresponds with the traditional idea of the structure of a dynamic system (see for example [66]), but with some important differences. First, this description uses auxiliary variables to keep track of the structural differences introduced by the composition of functions. This allows a degree of flexibility in how refined a view of the computational structure one considers "complete." Also, this description may have some slight differences in the labeling of nodes and edges compared with various descriptions in the literature. These differences will become important as new notions of partial structure are introduced in later sections. Here, we use rectangular nodes for graphs where the nodes represent *systems*, and the associated edges will represent *signals*. This convention will bridge well between the graphical representation of system structure and the typical engineering block diagram of a system, and it sometimes motivates the simplification in drawing edges as shown in Figure 21b), since every edge that leaves a node represents the same variable and carries the same label. Moreover, notice that nodes associated with the mechanisms that produce output variables are *terminal*, in that no edge ever leaves these nodes, while the nodes associated with input variables are *sources*, in that no edge ever arrives at these nodes. Although it is common for engineering diagrams to explicitly draw the edges associated with output variables and leave them "dangling," with no explicit terminal node, or to eliminate the input nodes and simply depict the input edges– also "dangling," our convention ensures that the diagram corresponds to a well defined *graph*, with every edge characterized by an ordered pair of nodes. Note also that state nodes, such as f_1 in Figure 21, may have self loops, although auxiliary nodes will not, and at times it will be convenient to partition the vertex set into groups corresponding to the input, state, auxiliary, and output mechanisms as $V(\mathcal{C}) = \{V_u(\mathcal{C}), V_x(\mathcal{C}), V_w(\mathcal{C}), V_y(\mathcal{C})\}$. Likewise, we may similarly partition the edge set as necessary.

We see, then, that knowing the complete structure $\mathcal{C}$ of a system is equivalent to knowing its state space realization, along with the composition structure with which these functions



(a) The complete computational structure $\mathcal{C}$ of the simple example specified by equation (45).



(b) A modified representation of the complete computational structure of the system specified by equation (45). Since the edges leaving a particular node will always represent the same variable, they have been combined to simplify the figure.

Figure 21: The complete computational structure of the state realization of a system is a graph demonstrating the dependency among all system variables; edges correspond to variables and nodes represent constitutive mechanisms that produce each variable. System outputs are understood to leave the corresponding terminal nodes, and system inputs arrive at the corresponding source nodes.

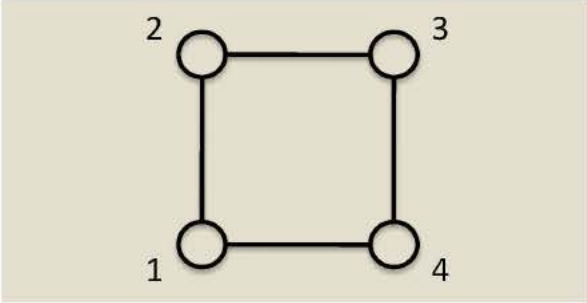
are represented, given by (f, g, h). We refer to this structure as *computational* because it reveals the dependencies among variables in the particular representation, or basis, that they are stored in and retrieved from memory. These specific, physical mechanisms that store and retrieve information, identified with $V_x(\mathcal{C})$, are interconnected with devices that transform variables, identified with $V_w(\mathcal{C})$, and with devices that interface with the system's external environment. These devices include sensors, identified with $V_u(\mathcal{C})$, and actuators, identified with $V_y(\mathcal{C})$, to implement the particular system behavior observed by the outside world through the *manifest* variables, u and y. Although other technologies very well may implement the same observed behavior via a different computational structure and a different representation of the *hidden* variables, x and w, $\mathcal{C}$ describes the structure of the actual system employing existing technologies as captured through a particular state description. In this sense, $\mathcal{C}$ is the complete architecture of the system, and often may be interpreted as the system's "physical layer." Importantly, it is often this notion of structure, or a very related concept, that is meant when discussing the "structure" of a system, as the next example illustrates.

Example: Graph Dynamical Systems As an example, we examine the computational structure of a graph dynamical system (GDS). Graph dynamical systems are finite alphabet, discrete time systems with dynamics defined in terms of the structure of an associated undirected graph. They have been employed in the study of various complex systems [67, 68], including

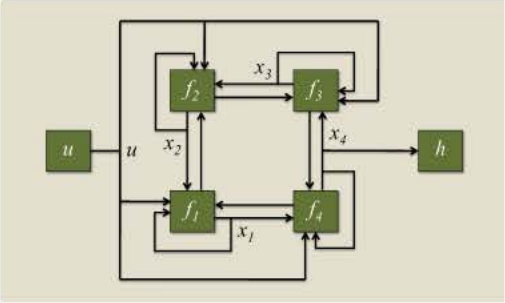
- Dynamical process on networks:
  - disease propagation over a social contact graph,
  - packet flow in cell phone communication,
  - urban traffic and transportation;
- Computational algorithms:
  - Gauss-Seidel,
  - gene annotation based on functional linkage networks,
  - transport computations on irregular grids;
- Computational paradigms related to distributed computing.

Here we observe that the computational structure of the graph dynamical system corresponds naturally with the system's underlying graph.

Given an undirected graph $\mathcal{G}$ with vertex set $V(\mathcal{G}) = \{1, 2, ..., n\}$, a GDS associates with each node i a state x_i that takes its values from a specified finite set $\mathcal{X}$. This state is then assigned a particular update function that updates its value according to the values of states associated with nodes adjacent to node i on $\mathcal{G}$. Notice that this restriction on the update function suggests that the update function for state i *depends* on states consistent with the structure of $\mathcal{G}$, indicating that the system's computational structure $\mathcal{C}$ should correspond to the adjacency structure of $\mathcal{G}$.



(a) Undirected graph $\mathcal{G}$ defining the sequential Graph Dynamical System (46).



(b) Computational structure $\mathcal{C}$ of the sequential Graph Dynamical System (46).

Figure 22: Graph Dynamical Systems are finite alphabet discrete time systems with dynamics defined in terms of the adjacency structure of a specified undirected graph. Here we note that the computational structure of the system reflects the structure of its underlying graph.

The distinction is made between a GDS that updates all of its states simultaneously, called a *parallel GDS*, and one that updates its states in a particular sequence, called a *sequential GDS*. The parallel GDS thus becomes an autonomous dynamical system, evolving its state according to its update function from some initial condition. The sequential GDS, on the other hand, can be viewed as a controlled system that receives a particular permutation of the node set $V(\mathcal{G})$ as input and then evolves its states accordingly.

To illustrate, consider the sequential GDS given by $\mathcal{G} = \{1, 2, 3, 4\}$ as indicated in Figure 22. Let $\mathcal{U} = \{1, 2, 3, 4\}$ and $\mathcal{X} = \mathcal{Y} = \{0, 1\}$, with update function given by

$$\begin{aligned} \begin{bmatrix} x_1[t+1] \\ x_2[t+1] \\ x_3[t+1] \\ x_4[t+1] \end{bmatrix} &= \begin{matrix} f_1(x[t], u[t]) \\ f_2(x[t], u[t]) \\ f_3(x[t], u[t]) \\ f_4(x[t], u[t]) \end{matrix} \\ y[t] &= x_4[t] \end{aligned} \tag{46}$$

where

$$f_i(x, u) = \begin{cases} x_i & u \neq i \\ (1 + x_i)(1 + x_{i-1})(1 + x_{i+1}) & u = i \end{cases} \tag{47}$$

and the arithmetic in (47) is taken modulo 2, while that in the subscript notation is taken modulo 4 (resulting in $x_5 \equiv x_1$, etc.). So, for example, the initial condition $x[0] = [0\ 0\ 0\ 0]^T$ with input sequence $u[t] = 1, 2, 3, 4, 1, 2, 3, 4, \ldots$ would result in the following periodic trajec-

tory:

$$\begin{array}{ll} x[1] = [1\ 0\ 0\ 0]^T & x[12] = [0\ 1\ 0\ 0]^T \\ \downarrow & \downarrow \\ x[2] = [1\ 0\ 0\ 0]^T & x[16] = [0\ 0\ 1\ 0]^T \\ \downarrow & \downarrow \\ x[3] = [1\ 0\ 1\ 0]^T & x[20] = [1\ 0\ 0\ 0]^T \\ \downarrow & \downarrow \\ x[4] = [1\ 0\ 1\ 0]^T & x[24] = [0\ 1\ 0\ 1]^T \\ \downarrow & \downarrow \\ x[8] = [0\ 0\ 0\ 1]^T & x[28] = [0\ 0\ 0\ 0]^T \end{array}$$

The computational structure of the system (46) follows immediately from the dependency among variables characterized by equation (47); Figure 22 illustrates $\mathcal{C}$ for this system. Notice that the structure of $\mathcal{G}$ is reflected in $\mathcal{C}$, where the undirected edges in $\mathcal{G}$ have been replaced by directed edges in both directions, and self-loops have appeared where applicable. Moreover, notice that $\mathcal{C}$ considers the explicit influence of the input sequence on the update computation, and explicitly identifies the system output. In this way, we see that the computational structure is a reflection of the natural structure of the sequential GDS.

Computational Structure of Linear Systems Linear systems represent an important special case of those described by (44). They arise naturally as the linearization of sufficiently smooth nonlinear dynamics near an equilibrium point or limit cycle, or as the fundamental dynamics of systems engineered to behave linearly under nominal operating conditions. In either case, knowing the structure of the relevant linear system is a critical first step to understanding that of the underlying nonlinear phenomena.

The general state description of a linear system is given by

$$\begin{array}{rcl} \dot{x} & = & Ax + \hat{A}w + Bu, \\ w & = & \bar{A}x + \tilde{A}w + \bar{B}u, \\ y & = & Cx + \bar{C}w + Du, \end{array} \tag{48}$$

where $A \in \mathbb{R}^{n\times n}$, $\hat{A} \in \mathbb{R}^{n\times l}$, $\bar{A} \in \mathbb{R}^{l\times n}$, $\tilde{A} \in \mathbb{R}^{l\times l}$, $B \in \mathbb{R}^{n\times m}$, $\bar{b} \in \mathbb{R}^{l\times m}$, $C \in \mathbb{R}^{p\times n}$, $\bar{C} \in \mathbb{R}^{p\times l}$, and $D \in \mathbb{R}^{p\times m}$. Note that $I - \tilde{A}$ is necessarily invertible, ensuring that the differentiability index of the system is zero. Nevertheless, the matrices are otherwise free.

As in the nonlinear case, it should be apparent that the auxiliary variables are superfluous in terms of characterizing the dynamic behavior of the system; this idea is made precise in the following lemma. Nevertheless, the auxiliary variables make a very important difference in terms of characterizing the system's complete computational structure, as illustrated by the subsequent example.

Lemma 2. *For any system (48) with intricacy $l > 0$, there exists a unique* minimal intricacy *realization (A_o, B_o, C_o, D_o) with $l = 0$ such that for every solution $(u(t), x(t), w(t), y(t))$ of (48), $(u(t), x(t), y(t))$ is a solution of (A_o, B_o, C_o, D_o).*

Proof. The result follows from the invertibility of $(I - \tilde{A})$. Solving for w and substituting into the equations of $\dot{x}$ and y then yields (A_o, B_o, C_o, D_o). □

Consider, for example, the system (48) with state matrices given by $D = 0$ and the following:

$$
\begin{aligned}
A &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \hat{A} = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \\
\bar{A} &= \begin{bmatrix} c_1 & 0 \\ 0 & c_2 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ a_1 & 0 \\ 0 & a_2 \end{bmatrix} \quad \tilde{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & e_1 & 0 & 0 & 0 & 0 & 0 & 0 \\ e_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\
B &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \bar{B}^T = \begin{bmatrix} 0 & 0 & 0 & 0 & b_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & b_2 & 0 & 0 \end{bmatrix} \\
C &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \bar{C} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}
\end{aligned}
\tag{49}
$$

This system has the complete computational structure $\mathcal{C}$ shown in Figure 23a. Here, because each auxiliary variable is defined as the simple product of a coefficient times another variable, we label the node corresponding to w_i in $\mathcal{C}$ with the appropriate coefficient rather than the generic label, g_i. Note that this realization has an intricacy of $l = 8$.

Suppose, however, that we eliminate the last six auxiliary variables, leading to an equivalent realization with intricacy $l = 2$. The state matrices then become

$$
\begin{aligned}
A &= \begin{bmatrix} a_1 & 0 \\ 0 & a_2 \end{bmatrix} \quad \hat{A} = \begin{bmatrix} 0 & e_1 \\ e_2 & 0 \end{bmatrix} \\
\bar{A} &= \begin{bmatrix} c_1 & 0 \\ 0 & c_2 \end{bmatrix} \qquad \tilde{A} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\
B &= \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix} \quad \bar{B}^T = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\
C &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \qquad \bar{C} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
\end{aligned}
\tag{50}
$$

with computational structure $\mathcal{C}$ as shown in Figure 23b. Similarly, we can find an equivalent realization with $l = 0$ given by

$$
A_o = \begin{bmatrix} a_1 & e_1 c_2 \\ e_2 c_1 & a_2 \end{bmatrix} \quad B_o = \begin{bmatrix} b_1 & 0 \\ 0 & b_2 \end{bmatrix} \quad C_o = \begin{bmatrix} c_1 & 0 \\ 0 & c_2 \end{bmatrix} \tag{51}
$$

and all other system matrices equal to zero. This realization, (51), is the minimal intricacy realization of both systems (49) and (50), and its complete computational structure $\mathcal{C}$ is given in Figure 23c. The equivalence between these realizations is easily verified by substitution.

Comparing the computational structures for different realizations of the same system, (49), (50), and (51), we note that the intricacy of auxiliary variables plays a critical role in suppressing or revealing system structure. Moreover, note that auxiliary variables can change the nature of which variables are manifest or hidden. In Figure 23 shaded regions indicate which variables, represented by edges, are hidden; manifest variables leave shaded regions while hidden variables are contained within them. Note that the minimal intricacy realization has no internal manifest variables, or, in other words, it has a single block of hidden variables (Figure 23c). Meanwhile, both w_1 and w_2 are manifest in the other realizations (Figures 23a, 23b) since $w_1 = y_1$ and $w_2 = y_2$, indicated by the "1" at their respective terminal nodes. This yields two distinct blocks of hidden variables, in either case, revealing the role intricacy of a realization can play characterizing its structure.

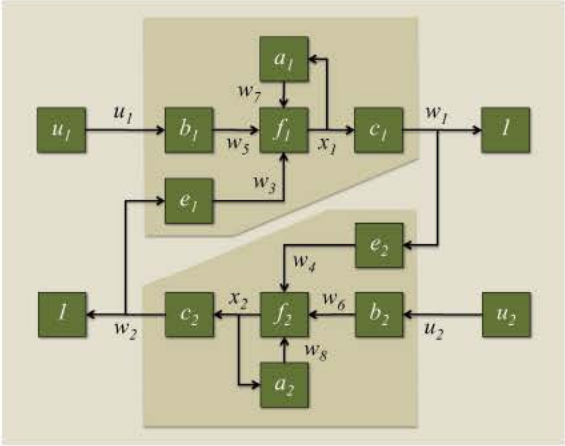
The complete computational structure of a system is thus a graphical representation of the dependency among input, state, auxiliary, and output variables that is in direct, one-to-one correspondence with the system's state realization, generalized to explicitly account for composition intricacy. All structural and behavioral information is fully represented by this description of a system. Nevertheless, this representation of the system can also be unwieldy for large systems with intricate structure.

4.5.2 Partial Structure Representations

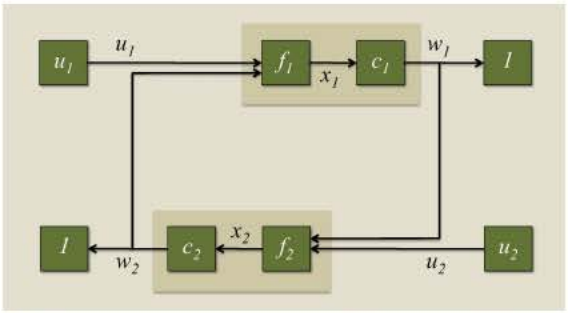
Complex systems are often characterized by intricate computational structure and complicated dynamic behavior. State descriptions and their corresponding complete computational structures accurately capture both the system's structural and dynamic complexity, nevertheless these descriptions themselves can be too complicated to convey an efficient understanding of the nature of the system. Simplified representations are then desirable.

One way to simplify the representation of a system is to restrict the structural information of the representation while maintaining a complete description of the system's dynamics. The most extreme example of this type of simplified representation is the transfer function of a single-input single-output linear time invariant (LTI) system. A transfer function completely specifies the system's input-output dynamics without retaining any information about the computational structure of the system. For example, consider the n^{th} order LTI single-input single-output system given by (A, b, c, d). It is well known that although the state description of the system completely specifies the transfer function, $G(s) = c(sI - A)^{-1}b + d$, the transfer function $G(s)$ has an infinite variety of state realizations, and hence computational structures, that all characterize the same input-output behavior. That is, the structural information in any state realization of the system is completely removed in the transfer function representation of the system, even though the dynamic (or behavioral) information about the system is preserved.

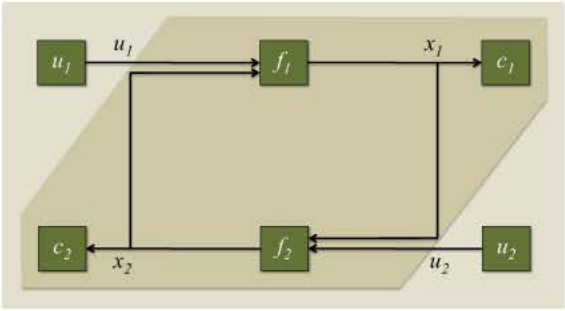
We use this power of a transfer function to obfuscate structural information to develop three distinct partial-structure representations of an LTI system: subsystem structure, signal structure, and the zero pattern of a (multiple input, multiple output) system's transfer function matrix. Later we will show how each of these representations has more structural



(a) The complete computational structure $\mathcal{C}$ of the linear system given by the state matrices (49) with intricacy $l = 8$.



(b) The complete computational structure $\mathcal{C}$ of the equivalent linear system with intricacy $l = 2$, specified by the realization (50).



(c) The complete computational structure $\mathcal{C}$ of the minimal intricacy ($l = 0$) realization for both systems above, characterized by (51).

Figure 23: The complete computational structure of a linear system characterized by realizations of differing intricacies. Edges within shaded regions represent *hidden* variables, while those outside shaded regions are *manifest* variables.

information than its successor, and we will precisely characterize the relationships among them.

Subsystem Structure One of the most natural ways to reduce the structural information in a system's representation is to partition the nodes of its computational structure into subsystems, then replace these subsystems with their associated transfer function. Each transfer function obfuscates the structure of its associated subsystem, and the remaining (partial) structural information in the system is the interconnection between transfer functions.

Subsystem structure refers to the appropriate decomposition of a system into constituent subsystems and the interconnection structure between these subsystems. Abstractly, it is the condensation graph of the complete computational structure graph, $\mathcal{C}$, taken with respect to a particular partition of $\mathcal{C}$ that identifies subsystems in the system. Such abstractions have been used in various ways to simplify the structural descriptions of complex systems [66, 69], for example by "condensing" strongly connected components or other groups of vertices of a graph into single nodes. Nevertheless, in this work we define a *particular* condensation graph as the subsystem structure of the system. We begin by characterizing the partitions of $\mathcal{C}$ that identify subsystems.

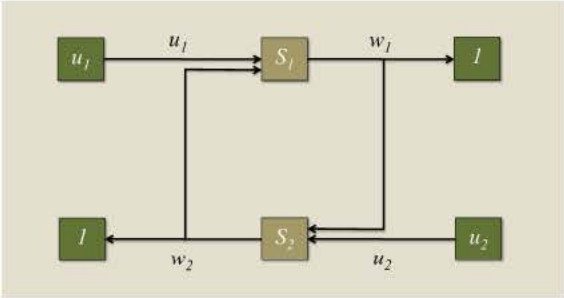
Definition 7. *Given a system G with realization (48) and associated computational structure $\mathcal{C}$, we say a partition of $V(\mathcal{C})$ is* admissible *if every edge in $E(\mathcal{C})$ between components of the partition represents a variable that is manifest, not hidden.*

For example, considering the system (51) with $V(\mathcal{C}) = \{u_1, f_1, c_1, c_2, f_2, u_2\}$. We see that the partition $\{(u_1), (f_1, c_1, c_2, f_2), (u_2)\}$ is admissible since the only edges between components are $u_1(u_1, f_1)$ and $u_2(u_2, f_2)$, representing the manifest variables u_1 and u_2. Notice that the shading in Figure 23c is consistent with this admissible partition. Alternatively, the partition $\{(u_1), (f_1, c_1), (c_2, f_2), (u_2)\}$ is not admissible for (51), since the edges $x_1(f_1, f_2)$ and $x_2(f_2, f_1)$ extend between components of the partition but represent variables x_1 and x_2 that are hidden, not manifest.

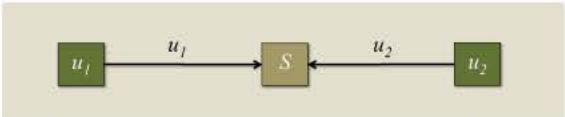
Although sometimes any aggregation, or set of fundamental computational mechanisms represented by vertices in $\mathcal{C}$, may be considered a valid subsystem, in this work a subsystem has a specific meaning. In particular, the variables that interconnect subsystems must be manifest, and thus subsystems are identified by the components of admissible partitions of $V(\mathcal{C})$. We adopt this convention to 1) enable the distinction between real subsystems vs. merely arbitrary aggregations of the components of a system, and 2) ensure that the actual subsystem architecture of a particular system is adequately reflected in the system's computational structure and associated realization, thereby ensuring that such realization is complete.

Definition 8. *Given a system G with realization (48) and associated computational structure $\mathcal{C}$, the system's* subsystem structure *is a condensation graph $\mathcal{S}$ of $\mathcal{C}$ with vertex set $V(\mathcal{S})$ and edge set $E(\mathcal{S})$ given by:*

- *$V(\mathcal{S}) = \{S_1, ...S_q\}$ are the elements of an admissible partition of $V(\mathcal{C})$ of maximal cardinality, and*



(a) The subsystem structure $\mathcal{S}$ of the linear system given by the state matrices (49) with intricacy $l = 8$. Note that the subsystem structure of the equivalent, but less intricate system given by equation (50) is exactly the same, indicated by the shaded regions in Figures 23a and 23b.



(b) The subsystem partial structure $\mathcal{S}$ of the dynamically equivalent, minimally intricate linear system, specified by the realization (51) corresponding to Figure 23c.

Figure 24: The subsystem structure of a linear system partitions vertices of the complete computational structure and condenses admissible groups of nodes into single subsystem nodes, shown in brown; nodes shown in green are those that correspond directly to unaggregated vertices of the complete computational structure, which will always be associated with mechanisms that generate manifest variables. Here, the subsystem structure corresponds to the interconnection of shaded regions in Figure 23. Comparing Figures 24a and 24b, we note that intricacy in a realization may be necessary to characterize meaningful subsystems and yield nontrivial subsystem structure.

- *$E(\mathcal{S})$ has an edge (S_i, S_j) if $E(\mathcal{C})$ has an edge from some component of S_i to some component of S_j.*

We label the nodes of $V(\mathcal{S})$ with the transfer function of the associated subsystem, which we also denote S_i, and the edges of $E(\mathcal{S})$ with the associated variable from $E(\mathcal{C})$.

Note that, like $\mathcal{C}$, the subsystem structure $\mathcal{S}$ is a graph with vertices that represent *systems* and edges that represent *signals*, or system variables. For example, Figure 24a illustrates the subsystem structure for both systems (49) and (50), shown in Figures 23a and 23b. Note that the subsystem structure of these systems' minimally intricate realization, (51), is quite different, with a single block rather than two blocks interconnected in feedback, as shown in Figure 24b. This illustrates the necessity of auxiliary variables to adequately describe the complete system structure.

Lemma 3. *The subsystem structure $\mathcal{S}$ of a system G, with complete computational structure $\mathcal{C}$, is unique.*

Proof. We prove by contradiction. Suppose the subsystem structure $\mathcal{S}$ of G is not unique. Then there are at least two distinct subsystem structures of G, which we will label $\mathcal{S}^1$ and $\mathcal{S}^2$. This implies there are two admissible partitions of $V(\mathcal{C})$, given by $V(\mathcal{S}^1)$ and $V(\mathcal{S}^2)$, such that $V(\mathcal{S}^1) \neq V(\mathcal{S}^2)$ and with equal cardinality, q. Note that by definition, q must be the maximal cardinality of any admisible partition of $V(\mathcal{C})$. To obtain a contradiction, we will construct another admissible partition, $V(\mathcal{S}^3)$, such that $|V(\mathcal{S}^3)| > q$.

Consider the following partition of $V(\mathcal{C})$ that is a refinement of both $V(\mathcal{S}^1)$ and $V(\mathcal{S}^2)$:

$$V(\mathcal{S}^3) = \{S^3 | S^3 \neq \emptyset; S^3 = S_i \cap S_j, S_i \in V(\mathcal{S}^1), S_j \in V(\mathcal{S}^2)\}.$$

Since $V(\mathcal{S}^1) \neq V(\mathcal{S}^2)$, then $|V(\mathcal{S}^3)| > q$, since the cardinality of the refinement must then be greater than that of $V(\mathcal{S}^1)$ or $V(\mathcal{S}^2)$. Moreover, note that the partition $V(\mathcal{S}^3)$ is admissible, since every edge of $\mathcal{C}$ between vertices associated with distinct components of $V(\mathcal{S}^3)$ corresponds with an edge of either $\mathcal{S}^1$ or $\mathcal{S}^2$, which are admissible. Thus, $V(\mathcal{S}^3)$ is an admissible partition of $V(\mathcal{C})$ with cardinality greater than q, which contradicts the assumption that $\mathcal{S}^1$ and $\mathcal{S}^2$ are both subsystem structures of G. □

The subsystem structure of a system reveals the way natural subsystems are interconnected, and it can be represented in other ways besides (but equivalent to) specifying $\mathcal{S}$. For example, one common way to identify this kind of subsystem architecture is to write the system as the linear fractional transformation (LFT) with a block diagonal "subsystem" component and a static "interconnection" component (see [70] for background on the LFT). For example, the system in Figure 24a can be equivalently represented by the feedback interconnection of a static system $N : \mathbb{U} \times \mathbb{W} \to \mathbb{Y} \times (\mathbb{U} \times \mathbb{W})$ and a block-diagonal dynamic system $S : \mathbb{U} \times \mathbb{W} \to \mathbb{W}$ given by

$$N = \left[\begin{array}{cc|cc} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{array}\right], \quad S = \left[\begin{array}{cc} S_1 & 0 \\ 0 & S_2 \end{array}\right], \tag{52}$$

where

$$S_1 : \left[\begin{array}{c} u_1 \\ w_2 \end{array}\right] \to w_1 \quad S_2 : \left[\begin{array}{c} w_1 \\ u_2 \end{array}\right] \to w_2 \tag{53}$$

In general, the LFT associated with $\mathcal{S}$ will have the form

$$N = \left[\begin{array}{cc} 0 & I \\ L & K \end{array}\right] \quad S = \left[\begin{array}{ccc} S_1 & 0 & ... \\ 0 & \ddots & \\ \vdots & 0 & S_q \end{array}\right] \tag{54}$$

where q is the number of distinct subsystems, and L and K are each binary matrices of the appropriate dimension (see Figure 25). Note that if additional output variables are present, besides the manifest variables used to interconnect subsystems, then the structure of N

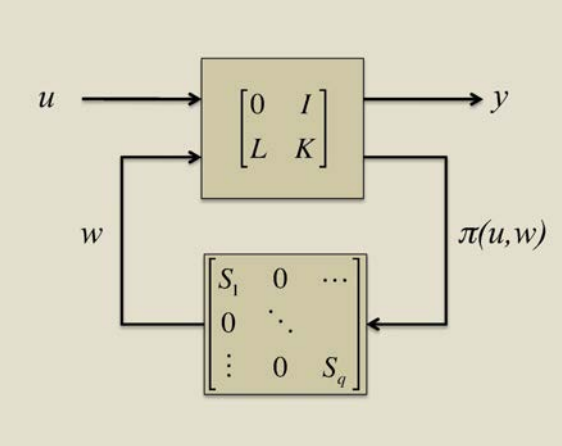


Figure 25: The subsystem structure of a system can always be represented by the lower linear fractional transformation of the static interconnection matrix N with a block diagonal transfer function matrix S. Note that $\pi(u, w)$ represents a permutation of a subset of the variables in the vector inputs, u, and manifest auxiliary variables, w, possibly with repetition of some variables if necessary.

and S above extend naturally. In any event, however, N is static and L and K are binary matrices.

The subsystem structure of any system is well defined although it may be trivial (a single internal block) if the system does not decompose naturally into an interconnection of subsystems, such as (51) in Figure 23c and Figure 24b. Note that $\mathcal{S}$ always identifies the most refined subsystem structure possible, and for systems with many interconnected subsystems, coarser representations may be obtained by aggregating various subsystems together and constructing the resulting condensation graph. These coarser representations effectively absorb some interconnection variables and their associated edges into the aggregated components, suggesting that such representations are the subsystem structure for less intricate realizations of the system, where some of the manifest auxiliary variables are removed, or at least left hidden. The subsystem structure is thus a natural partial description of system structure when the system can be decomposed into the interconnection of specific subsystems.

Signal Structure Another very natural way to partially describe the structure of a system is to characterize the direct causal dependence among each of its manifest variables; we will refer to this notion as the *signal structure*. This description of the structure of a system makes no attempt to cluster, or partition, the actual internal system states. As a result, it offers no information about the internal interconnection of subsystems, and signal structure can therefore be a very different description of system structure than subsystem structure.

Given a generalized linear system (48) with complete computational structure $\mathcal{C}$, we characterize its signal structure by considering its minimal intricacy realization (A_o, B_o, C_o, D_o). We assume without loss of generality that the outputs y are ordered in such a way that C_0

can be partitioned

$$C_o = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

where $C_{11} \in \mathbb{R}^{p_1 \times p_1}$ is invertible, with p_1 equal to the rank of C_o; $C_{12} \in \mathbb{R}^{p_1 \times (n-p_1)}$; $C_{21} \in \mathbb{R}^{(p-p_1)\times p_1}$; and $C_{22} \in \mathbb{R}^{(p-p_1)\times(n-p_1)}$. Note that if the outputs of the minimal intricacy realization do not result in C_{11} being invertible, then it is possible to reorder them so the first p_1 outputs correspond to independent rows of C_o; the states can then be renumbered so that C_{11} is invertible. One can show that such reordering of the outputs and states of the minimal intricacy realization only affects the ordering of the states and outputs of the original system; the graphical relationship of the computational structure is preserved.

The direct causal dependence among manifest variables is then revealed as follows. First, consider the state transformation $z = Tx$ given by

$$T = \begin{bmatrix} C_{11} & C_{12} \\ 0 & I \end{bmatrix}. \tag{55}$$

This transformation yields a system of the form

$$\begin{aligned} \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u \\ \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} &= \begin{bmatrix} I & 0 \\ C_2 & 0 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} D_1 \\ D_2 \end{bmatrix} u \end{aligned} \tag{56}$$

where $z \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $y_1 \in \mathbb{R}^{p_1}$, and $y_2 \in \mathbb{R}^{p-p_1}$. To simplify the exposition we will abuse notation and refer to the above system as (A, B, C, D), since there is little opportunity to confuse these matrices with those of the original system given in (48). In fact, the system (56) is simply a change of coordinates of the minimal intricacy realization (A_o, B_o, C_o, D_o), possibly with a reordering of the output and state variables. The direct causal dependence among manifest variables is then revealed by the dynamical structure function of $(A, B, \begin{bmatrix} I & 0 \end{bmatrix}, D_1)$.

The dynamical structure function of a class of linear systems was defined in [71] and discussed in [72, 73, 74, 75, 76, 77]. This representation of a linear system describes the direct causal dependence among a subset of state variables, and it will extend to characterize signal structure for the system in (56). We repeat and extend the derivation here to demonstrate its applicability to the system (56). Taking Laplace transforms and assuming zero initial conditions yields the following relationships

$$\begin{bmatrix} sZ_1 \\ sZ_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} U \tag{57}$$

where $Z(s)$ denotes the Laplace transform of $z(t)$, etc. Solving for Z_2 in the second equation and substituting into the first then yields

$$sZ_1 = W(s)Z_1 + V(s)U \tag{58}$$

where $W(s) = [A_{11} + A_{12}(sI - A_{22})^{-1}A_{21}]$ and $V(s) = [B_1 + A_{12}(sI - A_{22})^{-1}B_2]$. Let $\hat{D}(s)$ be the matrix of the diagonal entries of $W(s)$, yielding

$$Z_1 = Q(s)Z_1 + P(s)U \tag{59}$$

where $Q(s) = (sI - \hat{D}(s))^{-1}(W(s) - \hat{D}(s))$ and $P(s) = (sI - \hat{D}(s))^{-1}V(s)$. From (56) we note that $Z_1 = Y_1 - D_1U$, which, substituting into (59), then yields:

$$\begin{bmatrix} Y_1 \\ Y_2 \end{bmatrix} = \begin{bmatrix} Q(s) \\ C_2 \end{bmatrix} Y_1 + \begin{bmatrix} P(s) + (I - Q(s))D_1 \\ D_2 \end{bmatrix} U \tag{60}$$

The matrices $\begin{bmatrix} Q(s)^T & C_2^T \end{bmatrix}^T$ and $\begin{bmatrix} (P(s) + (I - Q(s))D_1)^T & D_2^T \end{bmatrix}^T$ we refer to as $\bar{Q}$ and $\bar{P}$, respectively. The matrices $(Q(s), P(s))$ are called the dynamical structure function of the system (56), and they characterize the dependency graph among manifest variables as indicated in Equation (60). We note a few characteristics of $(Q(s), P(s))$ that give them the interpretation of system structure, namely:

- $Q(s)$ is a square matrix of strictly proper real rational functions of the Laplace variable, s, with zeros on the diagonal. Thus, if each entry of y_1 were the node of a graph, $Q_{ij}(s)$ would represent the weight of a directed edge from node j to node i; the fact $Q_{ij}(s)$ is proper preserves the meaning of the directed edge as a *causal* dependency of y_i on y_j.
- Similarly, the entries of the matrix $[P(s) + (I - Q(s))D_1]$ carry the interpretation of causal weights characterizing the dependency of entires of y_1 on the m inputs, u. Note that when $D_1 = 0$, this matrix reduces to $P(s)$, which has *strictly* proper entries.

This leads naturally to the definition of signal structure.

Definition 9. *The* signal structure *of a system G, with realization (48) and equivalent realization (56), and with dynamical structure function $(Q(s), P(s))$ characterized by (59), is a graph $\mathcal{W}$, with a vertex set $V(\mathcal{W})$ and edge set $E(\mathcal{W})$ given by:*

- *$V(\mathcal{W}) = \{u_1, ..., u_m, y_{11}, ..., y_{1p_1}, y_{21}, ..., y_{2p_2}\}$, each representing a manifest signal of the system, and*
- *$E(\mathcal{W})$ has an edge from u_i to y_{1j}, u_i to y_{2j}, y_{1i} to y_{1j} or y_{1i} to y_{2_j} if the associated entry in $[P(s) + (I - Q(s))D_1]$, D_2, $Q(s)$, or C_{21} (as given in Equations (59) and (60)) is nonzero, respectively.*

We label the nodes of $V(\mathcal{W})$ with the name of the associated variable, and the edges of $E(\mathcal{W})$ with the associated transfer function entry from Equation (60).

Signal structure is fundamentally a different *type* of graph than either the computational or subsystem structure of a system because, unlike these other graphs, vertices of a system's signal structure represent *signals* rather than systems. Likewise, the edges of $\mathcal{W}$ represent *systems* instead of signals, as opposed to $\mathcal{C}$ or $\mathcal{S}$. We highlight these differences by using circular nodes in $\mathcal{W}$, in contrast to the square nodes in $\mathcal{C}$ or $\mathcal{S}$. The next example illustrates a system without notable subsystem structure and no apparent structural motif in its complete computational structure; nevertheless, it reveals a simple and elegant ring structure in the weak sense.

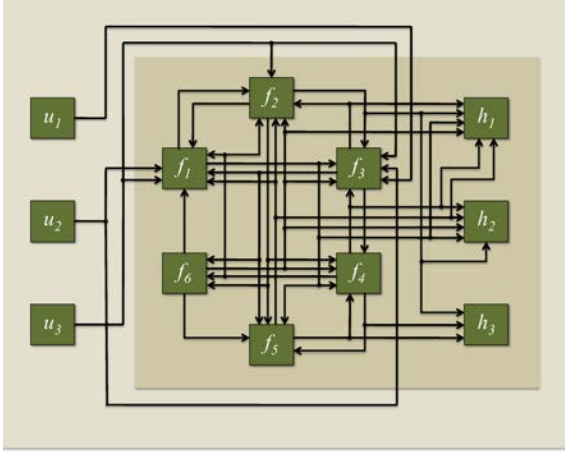


Figure 26: The complete computational structure $\mathcal{C}$, of the system (A_o, B_o, C_o, D_o) given by (61). Here, edge labels, u and x, and self loops on each node f_i have been omitted to avoid the resulting visual complexity. Edges associated with variables x, which are not manifest, are entirely contained within the shaded region (which also corresponds to the strong partial condensation shown in Figure 27a).

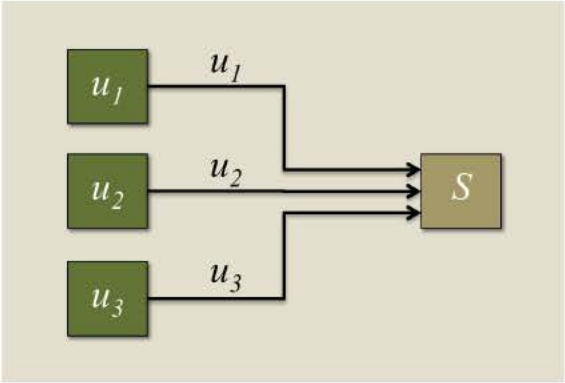
Example 3. Ring Signal Structure. *Systems with no apparent structure in any other sense can nevertheless be very structured in the weak sense. Consider the minimally intricate linear system, specified by the state-space realization* (A_o, B_o, C_o, D_o)*, where*

$$A_o = \frac{1}{12}\begin{bmatrix} -178 & 262 & -10 & -141 & -19 & 88 \\ -156 & 252 & -12 & -156 & -48 & 84 \\ -158 & 166 & -38 & -147 & -5 & 128 \\ -12 & 48 & -12 & -72 & -12 & 12 \\ -288 & 504 & 0 & -264 & -180 & 144 \\ 0 & 24 & 0 & -24 & -12 & -12 \end{bmatrix},$$

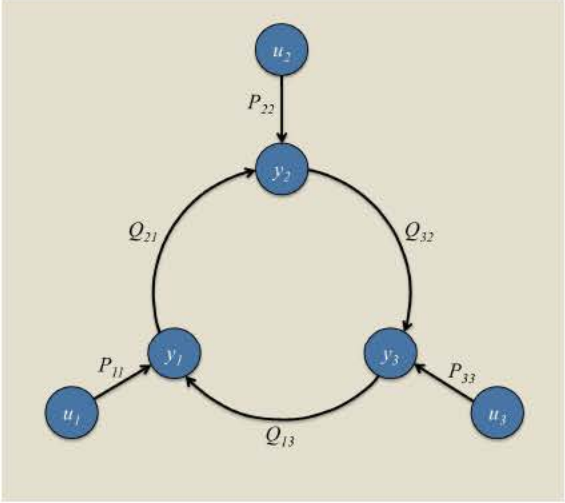
$$B_o = \frac{1}{4}\begin{bmatrix} 0 & -1 & 21 \\ 0 & 0 & 12 \\ -8 & 1 & 27 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$C_o = \begin{bmatrix} -1 & 4 & -1 & -2 & -1 & 1 \\ -12 & 21 & 0 & -11 & -5 & 6 \\ 0 & 2 & 0 & -2 & -1 & 0 \end{bmatrix},\ D_o = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{61}$$

We compute the signal structure by employing a change of coordinates on the state variables



(a) The subsystem structure, $\mathcal{S}$, of the system shown in Figures 26 and 27b; the system has no interconnection structure between subsystems because the system is composed of only a single subsystem; it does not meaningfully decompose into smaller subsystems.



(b) The signal structure, $\mathcal{W}$, of the system shown in Figures 26 and 27a. Note that, in contrast with $\mathcal{C}$ and $\mathcal{S}$, vertices of $\mathcal{W}$ represent manifest *signals* (characterized by round nodes), while edges represent *systems*.

Figure 27: Distinct notions of partial structure for the system (A_o, B_o, C_o, D_o) given by Equation (61). Observe that while the system is unstructured in the strong sense, it is actually quite structured in the weak sense. Moreover, although the subsystem structure is visible in the complete computational structure (Figure 26) as a natural condensation (note the shaded region), the signal structure is not readily apparent.

to find an equivalent realization of the form (56). The transformation

$$T = \begin{bmatrix} -1 & 4 & -1 & -2 & -1 & 1 \\ -12 & 21 & 0 & -11 & -5 & 6 \\ 0 & 2 & 0 & -2 & -1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{62}$$

results in the following realization

$$A = TA_0T^{-1} = \begin{bmatrix} -2 & 0 & 0 & 0 & 0 & -3 \\ 0 & -3 & 0 & -1 & 0 & 0 \\ 0 & 0 & -4 & 0 & 5 & 0 \\ 1 & 0 & 0 & -4 & 0 & 0 \\ 0 & 2 & 0 & 0 & -5 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 \end{bmatrix}$$

$$B = TB_0 = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 6 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad C = C_0T^{-1} = \begin{bmatrix} I_3 & 0 \end{bmatrix}, \qquad D = D_0 = [0]. \tag{63}$$

The dynamical structure function of the system, (Q, P), then becomes

$$Q = \begin{bmatrix} 0 & 0 & \frac{-3}{s^2+3s+2} \\ \frac{-1}{s^2+7s+12} & 0 & 0 \\ 0 & \frac{10}{s^2+9s+20} & 0 \end{bmatrix},$$

$$P = \begin{bmatrix} \frac{2}{s+2} & 0 & 0 \\ 0 & \frac{3}{s+3} & 0 \\ 0 & 0 & \frac{6}{s+4} \end{bmatrix} \tag{64}$$

which yields the signal structure, $\mathcal{W}$, as shown in Figure 27b. Notice that although the complete computational structure and subsystem structure do not characterize any meaningful interconnection patterns, the system is nevertheless structured and organized in a very concrete sense. In particular, the outputs y_1, y_2, and y_3 form a cyclic dependency, and each causally depends on a single input $u_i, i = 1, 2, 3,$, respectively.

Zero Pattern of the Transfer Function Matrix The weakest notion of structure exhibited by a system is the pattern of zeros portrayed in its transfer function matrix, where "zero" refers to the value of the particular transfer function element, not a transmission zero of the system. Like signal structure, this type of structure is particularly meaningful for multiple-input multiple-output systems, and, like signal structure, the corresponding graphical representation reflects the dependance of system output variables on system input variables. Thus, nodes of the graph will be signals, represented by circular nodes, and the edges of the graph will represent systems, labeled with the corresponding transfer function element; a zero element thus corresponds to the absence of an edge between the associated system input and output. Formally, we have the following definition.

Definition 10. *The* zero pattern of the transfer function *of a system G is a graph $\mathcal{Z}$, with a vertex set $V(\mathcal{Z})$ and edge set $E(\mathcal{Z})$ given by:*

- $V(\mathcal{Z}) = \{u_1, ..., u_m, y_1, ..., y_p\}$*, each representing a manifest signal of the system, and*
- $E(\mathcal{Z})$ *has an edge from u_i to y_j if G_{ji} is nonzero.*

We label the nodes of $V(\mathcal{Z})$ with the name of the associated variable, and the edges of $E(\mathcal{Z})$ with the associated element from the transfer function $G(s)$.

Unlike signal structure, note that the zero pattern of the transfer function matrix describes the closed-loop dependency of an output variable on a particular input variable, not its *direct* dependence. As a result, the graph is necessarily bipartite, and all edges will begin at an input node and terminate at an output node; no edges will illustrate dependencies between output variables. For example, the zero pattern of the transfer function for the system in Example 3, is shown in Figure 28, with transfer function $G(s) =$

$$\begin{bmatrix} \frac{n_1(s)}{d(s)} & \frac{-90(s+4)}{d(s)} & \frac{-18(s^3+12s^2+47s+60)}{d(s)} \\ \frac{-2(s^3+10s^2+29s+20)}{d(s)} & \frac{3(s^5+16s^4+97s^3+274s^2+352s+160)}{d(s)} & \frac{18(s+5)}{d(s)} \\ \frac{-20(s+1)}{d(s)} & \frac{30(s^3+7s^2+14s+8)}{d(s)} & \frac{n_2(s)}{d(s)} \end{bmatrix}, \tag{65}$$

where $d(s) = s^6 + 19s^5 + 145s^4 + 565s^3 + 1170s^2 + 1220s + 450$, $n_1(s) = 2(s^5 + 17s^4 + 111s^3 + 343s^2 + 488s + 240)$, and $n_2(s) = 6(s^5 + 15s^4 + 85s^3 + 225s^2 + 274s + 120)$. Although the *direct* dependence, given by the signal structure, is cyclic (Figure 27b), there is a path from every input to every output that does not cancel. Thus, the zero pattern of the transfer function is fully connected, corresponding to the full transfer function matrix in Equation (65).

It is important to understand that the zero pattern of a transfer function does not necessarily describe the flow of information between inputs and outputs. The presence of a zero in the $(i, j)^{\text{th}}$ location simply indicates that the input-output response of the system results in the i^{th} output having no dependence on the j^{th} input. Such an effect, however, could be the result of certain internal cancellations and does not suggest, for example, that there is no path in the complete computational structure from the j^{th} input to the i^{th} output. Thus, for example, a diagonal transfer function matrix does not imply the system is decoupled.

The next example demonstrates that a fully coupled system may, nevertheless, have a diagonal transfer function, even when the system is minimal in both intricacy and order. That is, the internal cancellations necessary to generate the diagonal zero pattern in the transfer

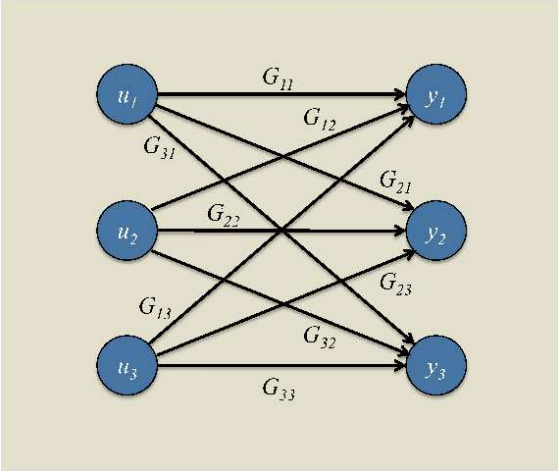


Figure 28: The zero pattern of the transfer function for the system in Example 3. Note that vertices of the zero pattern are manifest *signals*, distinguished in this work by circular nodes similar to those of signal structure. Nevertheless, this system has a fully connected (and thus unstructured) zero pattern, while its signal structure (shown in Figure 27b) exhibits a definite ring structure.

function while being fully coupled do not result in a loss of controllability or observability. Thus, the zero pattern of a transfer function only describes the input-output dependencies of the system and does not imply anything about the internal information flow or communication structure. This fact is especially important in the context of decentralized control, where the focus is often on shaping the zero pattern of a controller's transfer function given a particular zero pattern in the transfer function of the system to be controlled.

Example 4. Diagonal Transfer Function $\neq$ Decoupled. *Consider a system, G, with the following minimal intricacy realization, (A_o, B_o, C_o, D_o):*

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} &= \begin{bmatrix} -5 & 1 \\ 2 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 2 & 1 \\ 4 & -1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \\ \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} &= \begin{bmatrix} 1 & 1 \\ -4 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \end{aligned} \tag{66}$$

It is easy to see from (A_o, B_o, C_o, D_o) that this system has a fully connected complete computational structure. Moreover, one can easily check that the realization is controllable and observable, and thus of minimal order. Nevertheless, its transfer function is diagonal, given by

$$G(s) = \begin{bmatrix} \frac{6}{s+3} & 0 \\ 0 & \frac{-6}{s+6} \end{bmatrix}. \tag{67}$$

4.5.3 Relationships Among Representations of Structure

In this section, we explore the relationships between the four representations of structure defined above. What we will find is that some representations of structure are more informative

than others. Next, we will discus a specific class of systems in which the four representations of structure can be ranked by information content (with one representation encapsulating all the information contained in another representation of structure). Outside this class of systems, however, we will demonstrate that signal structure and subsystem structure have fundamental differences in addition to those arising from the graphical conventions of circular versus square nodes, etc. Signal and subsystem structure provide alternative ways of discussing a system's structure without requiring the full detail of a state-space realization or the abstraction imposed by a transfer function. Understanding the relationships between these representations then enables the study of new kinds of research problems that deal with the realization, reconstruction, and approximation of system structure (where structure can refer to the four representations defined so far or other representations of structure not mentioned in this work). The final section gives a discussion of such future research.

Different representations of structure contain different kinds of structural information about the system. For example, the complete computational structure details the structural dependencies among fundamental units of computation. Using complete computational structure to model system structure requires knowledge of the parameters associated with each fundamental unit of computation. Partial representations of structure such as signal and subsystem structure do not require knowledge of such details in their description. Specifically, subsystem structure essentially aggregates units of computation to form subsystems and models the closed-loop transfer function of each subsystem. Signal structure models the SISO transfer functions describing direct causal dependencies between outputs and inputs of some of the fundamental units of computation that happen to be manifest. Zero pattern structure models the closed-loop dependencies of system outputs on inputs. Thus, complete computational structure appears to be the most demanding or information-rich description of system structure. Indeed, this intuition is made precise with the following result:

Theorem 1. *Suppose a complete computational structure has minimal intricacy realization* (A_o, B_o, C_o, D_o) *with*

$$C_0 = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

and C_{11} *invertible. Then the complete computational structure specifies a unique subsystem, signal, and zero pattern structure.*

Proof. Let $\mathcal{C}$ be a computational structure with minimal intricacy realization (A_o, B_o, C_o, D_o) with

$$C_0 = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix},$$

C_{11} invertible. By Lemma 3, the subsystem structure $\mathcal{S}$ is unique. Since C_{11} is invertible, we see that equations (58) and (60) imply that the minimal intricacy realization uniquely specifies the dynamical structure function of the system. By definition, the signal structure is unique. Finally, write the generalized state-space realization of $\mathcal{C}$ as

$$(\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}) = \left(\begin{bmatrix} A & \hat{A} \\ \bar{A} & \tilde{A} \end{bmatrix}, \begin{bmatrix} B \\ \bar{B} \end{bmatrix}, \begin{bmatrix} C & \bar{C} \end{bmatrix}, \mathbf{D} \right).$$

The uniqueness of the zero pattern structure follows from its one-to-one correspondence with the transfer function $G(s) = C_o(sI - A_o)^{-1}B_o + D_o$ which can also be expressed as

$$(C + \bar{C}(I - \tilde{A})^{-1}\bar{A})(sI - A - \hat{A}(I - \tilde{A})^{-1}\bar{A})^{-1}(B + \bar{B}(I - \tilde{A})^{-1}\bar{A}).$$

□

It is well known that a transfer function $G(s)$ can be realized using an infinite number of state-space realizations. Without additional assumptions, e.g. full state feedback, it is impossible to uniquely associate a single state-space realization with a given transfer function. On the other hand, a state space realization specifies a unique transfer function. In this sense, a transfer function contains less information than the state space realization.

Similarly, subsystem, signal, and zero pattern structure can be realized using multiple complete computational structures. Without additional assumptions, it is impossible to associate a unique complete computational structure with a given subsystem, signal, or zero pattern structure. Theorem 1 shows that a complete computational structure specifies a unique subsystem, signal, and zero pattern structure. In this sense, a complete computational structure is a more informative description of system structure than subsystem, signal and zero pattern structure. The next result has a similar flavor and follows directly from the one-to-one correspondence of a system's transfer function with its zero pattern structure.

Theorem 2. *Every subsystem structure or signal structure specifies a unique zero pattern structure.*

Proof. Consider the LFT representation $\mathcal{F}(N, S)$ of a subsystem structure $\mathcal{S}$; write

$$N = \left[\begin{array}{c|c} 0 & I \\ \hline L & K \end{array}\right]$$

as in equation (54). The linear fractional transform gives the input-output map, i.e. the transfer function. Thus, $G(s) = (I - SK)^{-1}SL$.

Similarly, using the dynamical structure representation of the signal structure $\mathcal{W}$ given in equation (60), we can solve for the transfer function

$$G(s) = \left(I - \begin{bmatrix} Q(s) & 0 \\ C_2 & 0 \end{bmatrix}\right)^{-1} \begin{bmatrix} P(s) + (I - Q(s))D_1 \\ D_2 \end{bmatrix}$$

The result follows from the definition of zero pattern structure. □

The relationship between subsystem structure and signal structure is not so straightforward. Nevertheless, subsystem structure does specify a unique signal structure for a class of systems, namely systems with subsystem structure composed of single output (SO) subsystems and where every manifest variable is involved in subsystem interconnection. For this class of systems, subsystem structure is a richer description of system structure than signal structure.

Single Output Subsystem Structure and Signal Structure

Theorem 3. *Let $\mathcal{S}$ be a SO subsystem structure with LFT representation $\mathcal{F}(N, S)$. Suppose*

$$N = \left[\begin{array}{c|c} 0 & I \\ \hline L & K \end{array}\right],$$

where $\begin{bmatrix} L & K \end{bmatrix}$ has full column rank. Then $\mathcal{S}$ uniquely specifies the signal structure of the system.

Proof. The dynamics of N and S satisfy

$$\begin{aligned} Y &= [0]\,U + [I]\,Y && (68)\\ \pi &= LU + KY && (69)\\ Y &= S\pi && (70) \end{aligned}$$

Combining the second and third equation, we get

$$Y = S\pi = S\begin{bmatrix} K & L \end{bmatrix}\begin{bmatrix} Y \\ U \end{bmatrix}.$$

Since $\mathcal{S}$ is a SO subsystem structure, the entries of S describe direct causal dependencies among manifest variables involved in interconnection. Since $\begin{bmatrix} L & K \end{bmatrix}$ has full column rank and is a binary matrix, this means that each manifest variable is used at least once in interconnection. Thus, S describes the direct causal dependencies between all manifest variables of the system and specifies the signal structure of the system. □

Notice that for the class of systems described above, the four representations of structure can be ordered in terms of information content. Theorem 1 shows that the complete computational structure uniquely specifies all the other representations of structure and thus is the most informative of the four. By Theorem 3 and Theorem 2 respectively, subsystem structure uniquely specifies the signal structure and zero pattern structure of the system and thus is the second most informative. Similarly, signal structure is the third most informative and zero pattern structure is the least informative of the four representations of structure.

We note that the converse of Theorem is also true, namely if the subsystem structure of a system specifies a unique signal structure then the subsystem structure is a SO subsystem structure where every manifest variable is an interconnection variable. The proof is simple and follows from the result of the next subsection. We also provide several examples that show how a multiple output subsystem structure can be consistent with multiple signal structures - these all will serve to illustrate the general relationship between subsystem and signal structure outside of the class of systems mentioned above.

The Relationship Between Subsystem and Signal Structure Subsystem structure and signal structure are fundamentally different descriptions of system structure. In general, subsystem structure does not encapsulate the information contained in signal structure. Signal structure describes direct causal dependencies between manifest variables of the system. Subsystem structure describes closed loop dependencies between manifest variables involved

in the interconnection of subsystems. Both representations reveal different perspectives of a system's structure. The next result makes this relationship between subsystem and signal structure precise.

Theorem 4. *Given a system G, let $\mathcal{F}(N,S)$ be the LFT representation of a subsystem structure $\mathcal{S}$. In addition, let the signal structure of the system G be denoted as in equation (60). Let $Y(S_i)$ denote the outputs associated with subsystem S_i. Define*

$$[Q_{int}(s)]_{ij} \equiv \begin{cases} \bar{Q}_{ij}(s) & y_i, y_j \in Y(S_k), S_k \text{ a subsystem in } \mathcal{S} \\ 0 & \text{otherwise,} \end{cases}$$

and $Q_{ext} \equiv \bar{Q}(s) - Q_{int}(s)$. Then the signal structure and subsystem structure are related in the following way:

$$S \left[\begin{array}{c|c} L & K \end{array}\right] = (I - Q_{int})^{-1} \left[\begin{array}{c|c} \bar{P} & Q_{ext} \end{array}\right] \tag{71}$$

Proof. Examining relation (71), observe that the ij^{th} entry of the left hand side describes the closed loop causal dependency from the j^{th} entry of $\left[\begin{array}{cc} U^T & Y^T \end{array}\right]^T$ to Y_i. By closed loop, we mean that they do not describe the internal dynamics of each subsystem, e.g. the direct causal dependencies among outputs of a single subsystem. Thus, these closed loop causal dependencies are obtained by solving out the intermediate direct causal relationships, i.e. the entries in Q_{int}. Notice that the right hand side of (71) also describes the closed loop map from $\left[\begin{array}{cc} U^T & Y^T \end{array}\right]^T$ to Y, and in particular the ij^{th} entry of

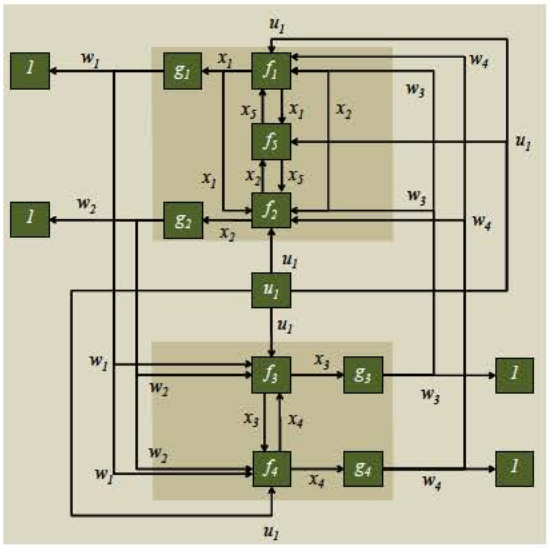
$$(I - Q_{int})^{-1} \left[\begin{array}{c|c} \bar{P} & Q_{ext} \end{array}\right]$$

describes the closed loop causal dependency from the jth entry of $\left[\begin{array}{cc} U & Y \end{array}\right]^T$ to Y_i. □

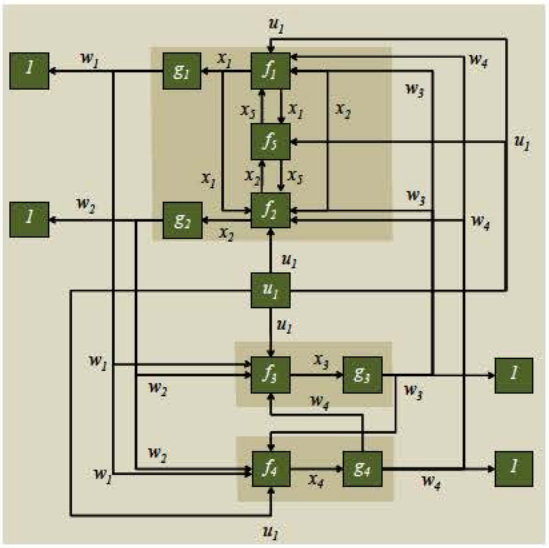
As a special case, notice that for SO subsystem structures, Q_{int} becomes the zero matrix and that for subsystem structures with a single subsystem, S becomes the system transfer function, L becomes the identity matrix, $Q_{int} = \bar{Q}$, and Q_{ext} and K are both zero matrices. The primary import of this result is that a single subsystem structure can be consistent with two or more signal structures and that a single signal structure can be consistent with two or more subsystem structures. Consider the following examples:

Example 5. A Signal Structure consistent with two Subsystem Structures
In this example, we will show how a signal structure can be consistent with two subsystem structures. To do this we construct two different generalized state-space realizations that yield the same minimal intricacy realization but different admissible partitions (see Definition 7). The result is two different subsystem structures that are associated with the same signal structure. First, we consider the complete computational structure $\mathcal{C}_1$ with generalized state-space realization

$$(\mathbf{A_1}, \mathbf{B_1}, \mathbf{C_1}, \mathbf{D_1}) = \left(\left[\begin{array}{cc} A_1 & \hat{A}_1 \\ \bar{A}_1 & \tilde{A}_1 \end{array}\right], \left[\begin{array}{c} B_1 \\ \bar{B}_1 \end{array}\right], \left[\begin{array}{cc} C_1 & \bar{C}_1 \end{array}\right], D_1 \right)$$



(a) The complete computational structure $\mathcal{C}_1$ with generalized state-space realization $(\mathbf{A_1}, \mathbf{B_1}, \mathbf{C_1}, \mathbf{D_1})$. We have omitted the self-loops on each of the f vertices for the sake of visual clarity.



(b) The complete computational structure $\mathcal{C}_2$ with generalized state-space realization $(\mathbf{A_2}, \mathbf{B_2}, \mathbf{C_2}, \mathbf{D_2})$. We have omitted the self-loops on each of the f vertices for the sake of visual clarity.

Figure 29: The complete computational structures of two systems that differ only by a slight rearrangement of their state-space dynamics. The rearrangement results in two different subsystem structure representations. Nonetheless, the resulting minimal intricacy realization for both systems is the same, implying that both systems have an identical signal structure.

where

$$A_1 = \begin{bmatrix} -4 & 1 & 0 & 0 & 1 \\ 1 & -7 & 0 & 0 & 3 \\ 0 & 0 & -6 & 0 & 0 \\ 0 & 0 & 0 & -6 & 0 \\ 1 & 2 & 0 & 0 & -10 \end{bmatrix}, \quad \hat{A}_1 = \begin{bmatrix} 0 & 0 & 2 & 1 \\ 0 & 0 & 2 & 1 \\ 2 & 1 & 0 & 1 \\ 1 & 2 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\bar{A}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \qquad \tilde{A}_1 = [0]_4,$$

$$B_1 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T, \qquad \bar{B}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix},$$

$$C_1 = \mathbf{0_{4\times 5}}, \qquad \bar{C}_1 = I_4,$$

and $D_1 = \mathbf{0_{4\times 1}}$. *Figure 29a shows the computational structure* $\mathcal{C}_1$. *Next consider the complete computational structure* $\mathcal{C}_2$ *with generalized state-space realization*

$$(\mathbf{A_2}, \mathbf{B_2}, \mathbf{C_2}, \mathbf{D_2}) = \left(\begin{bmatrix} A_2 & \hat{A}_2 \\ \bar{A}_2 & \tilde{A}_2 \end{bmatrix}, \begin{bmatrix} B_2 \\ \bar{B}_2 \end{bmatrix}, \begin{bmatrix} C_2 & \bar{C}_2 \end{bmatrix}, D_2 \right)$$

where

$$A_2 = \begin{bmatrix} -4 & 1 & 0 & 0 & 1 \\ 1 & -7 & 0 & 0 & 3 \\ 0 & 0 & -6 & 1 & 0 \\ 0 & 0 & 2 & -6 & 0 \\ 1 & 2 & 0 & 0 & -10 \end{bmatrix}, \quad \hat{A}_2 = \begin{bmatrix} 0 & 0 & 2 & 1 \\ 0 & 0 & 2 & 1 \\ 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\bar{A}_2 = \bar{A}_1, \qquad \tilde{A}_2 = [0]_4,$$

$$B_2 = B_1 = \mathbf{1_{5\times 1}}, \qquad \bar{B}_2 = \bar{B}_1 = \mathbf{0_{4\times 1}},$$

$$C_2 = C_1, \qquad \bar{C}_2 = \bar{C}_1,$$

and $D_2 = D_1$. *Figure 29b shows the computational structure* $\mathcal{C}_2$. *The difference between these two computational structures is evident more in the subsystem structure representation of the system - note how replacing* A_1 *with* A_2, *essentially externalizes internal dynamics. The result is that* $\mathcal{C}_2$ *admits a subsystem structure* $\mathcal{S}_2$ *which divides one of the subsystems of* $\mathcal{S}_1$ *into two subsystems. This is more apparent in the LFT representations of* $\mathcal{S}_1$ *and* $\mathcal{S}_2$; *the LFT representation of* $\mathcal{S}_1$ *is given by* $\mathcal{F}(N_1, S_1) =$

$$N_1 = \left[\begin{array}{c|cccc} \mathbf{0_{4\times 1}} & \multicolumn{4}{c}{\mathbf{I_4}} \\ \hline 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

and

$$S_1 = \begin{bmatrix} S_{11} & 0 & 0 \\ 0 & S_{12} & 0 \\ 0 & 0 & S_{13} \end{bmatrix},$$

with S_{11}, S_{12}, S_{13} *given by*

$$\begin{bmatrix} \frac{2(s^2+18s+76)}{s^3+21s+130s+234} & \frac{s^2+18s+76}{s^3+21s+130s+234} & \frac{s^2+19s+86}{s^3+21s^2+130s+234} \\ \frac{2(s^2+15s+52)}{s^3+21s^2+130s+234} & \frac{s^2+15s+52}{s^3+21s^2+130s+234} & \frac{(13+s)(s+5)}{s^3+21s^2+130s+234} \end{bmatrix},$$

$$\begin{bmatrix} \frac{2}{s+6} & \frac{1}{s+6} & \frac{1}{s+6} & \frac{1}{s+6} \end{bmatrix},$$

$$\begin{bmatrix} \frac{1}{s+6} & \frac{2}{s+6} & \frac{2}{s+6} & \frac{1}{s+6} \end{bmatrix}$$

respectively.

The LFT representation of $\mathcal{S}_2$ is represented as the LFT $\mathcal{F}(N_2, S_2) =$ where

$$N_2 = \left[\begin{array}{c|cccc} \mathbf{0_{4\times 1}} & \multicolumn{4}{c}{\mathbf{I_4}} \\ \hline 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

and

$$S_2 = \begin{bmatrix} S_{21} & 0 \\ 0 & S_{22} \end{bmatrix},$$

$$S_{21} = \begin{bmatrix} \frac{2(s^2+18s+76)}{s^3+21s^2+130s+234} & \frac{s^2+18s+76}{s^3+21s^2+130s+234} & \frac{s^2+19s+86}{s^3+21s^2+130s+234} \\ \frac{2(s^2+15s+52)}{s^3+21s^2+130s+234} & \frac{s^2+15s+52}{s^3+21s^2+130s+234} & \frac{(13+s)(s+5)}{s^3+21s^2+130s+234} \end{bmatrix},$$

$$S_{22} = \begin{bmatrix} \frac{2s+13}{s^2+12s+34} & \frac{s+8}{s^2+12s+34} & \frac{7+s}{s^2+12s+34} \\ \frac{s+10}{s^2+12s+34} & \frac{2(7+s)}{s^2+12s+34} & \frac{s+8}{s^2+12s+34} \end{bmatrix}.$$

However, if we consider the minimal intricacy realizations of $\mathcal{C}_1, \mathcal{C}_2$ we get the same state-space realization (A_o, B_o, C_o, D_o) with

$$A_o = \begin{bmatrix} -4 & 1 & 2 & 1 & 1 \\ 1 & -7 & 2 & 1 & 3 \\ 2 & 1 & -6 & 1 & 0 \\ 1 & 2 & 2 & -6 & 0 \\ 1 & 2 & 0 & 0 & -10 \end{bmatrix}, \quad B_o = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$C = \begin{bmatrix} I_4 & \mathbf{0_{4\times 1}} \end{bmatrix}$$

The signal structure of the system is thus specified by the dynamical structure function $(Q, P)(s)$, with

$$Q(s) = \begin{bmatrix} 0 & \frac{12+s}{s^2+14s+39} & \frac{2(s+10)}{s^2+14s+39} & \frac{s+10}{s^2+14s+39} \\ \frac{13+s}{s^2+17s+64} & 0 & \frac{2(s+10)}{s^2+17s+64} & \frac{s+10}{s^2+17s+64} \\ \frac{2}{s+6} & \frac{1}{s+6} & 0 & \frac{1}{s+6} \\ \frac{1}{s+6} & \frac{2}{s+6} & \frac{2}{s+6} & 0 \end{bmatrix}$$

$$P(s) = \begin{bmatrix} \frac{11+s}{s^2+14s+39} \\ \frac{13+s}{s^2+17s+64} \\ \frac{1}{s+6} \\ \frac{1}{s+6} \end{bmatrix}$$

Example 6. A Subsystem Structure consistent with two Signal Structures
Now we consider a subsystem structure with multiple signal structures. Recalling the discussion above, subsystem structure describes the closed loop *causal dependencies between manifest interconnection variables and signal structure specifies direct causal dependencies between*

manifest variables. When it is impossible to determine these direct causal dependencies by inspection from the closed loop causal dependencies in a subsystem structure representation, then there can be multiple signal structures that are consistent with the subsystem structure.

Reconsider $\mathcal{S}_2$. The LFT is given by $\mathcal{F}(N_2, S_2)$ where

$$N_2 = \left[\begin{array}{c|cccc} \mathbf{0_{4\times 1}} & \multicolumn{4}{c}{\mathbf{I_4}} \\ \hline 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{array}\right]$$

and

$$S_2 = \begin{bmatrix} S_{21} & 0 \\ 0 & S_{22} \end{bmatrix},$$

with S_{21} and S_{22} given by

$$\begin{bmatrix} \frac{2(s^2+18s+76)}{s^3+21s^2+130s+234} & \frac{s^2+18s+76}{s^3+21s^2+130s+234} & \frac{s^2+19s+86}{s^3+21s^2+130s+234} \\ \frac{2(s^2+15s+52)}{s^3+21s^2+130s+234} & \frac{s^2+15s+52}{s^3+21s^2+130s+234} & \frac{(13+s)(s+5)}{s^3+21s^2+130s+234} \end{bmatrix},$$

$$\begin{bmatrix} \frac{2s+13}{s^2+12s+34} & \frac{s+8}{s^2+12s+34} & \frac{7+s}{s^2+12s+34} \\ \frac{s+10}{s^2+12s+34} & \frac{2(7+s)}{s^2+12s+34} & \frac{s+8}{s^2+12s+34} \end{bmatrix}$$

respectively.

If we consider the relation

$$S_2 \begin{bmatrix} K & L \end{bmatrix} = (I - Q_{int})^{-1} \begin{bmatrix} Q_{ext} & P \end{bmatrix},$$

we can take $(Q, P)(s)$ to equal

$$Q_1(s) = \begin{bmatrix} 0 & \frac{12+s}{s^2+14s+39} & \frac{2(s+10)}{s^2+14s+39} & \frac{s+10}{s^2+14s+39} \\ \frac{13+s}{s^2+17s+64} & 0 & \frac{2(s+10)}{s^2+17s+64} & \frac{s+10}{s^2+17s+64} \\ \frac{2}{s+6} & \frac{1}{s+6} & 0 & \frac{1}{s+6} \\ \frac{1}{s+6} & \frac{2}{s+6} & \frac{2}{s+6} & 0 \end{bmatrix}$$

$$P_1(s) = \begin{bmatrix} \frac{11+s}{s^2+14s+39} \\ \frac{13+s}{s^2+17s+64} \\ \frac{1}{s+6} \\ \frac{1}{s+6} \end{bmatrix}$$

with

$$Q_{int} \equiv \begin{bmatrix} 0 & \frac{12+s}{s^2+14s+39} & 0 & 0 \\ \frac{13+s}{s^2+17s+64} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{s+6} \\ 0 & 0 & \frac{2}{s+6} & 0 \end{bmatrix}$$

and $Q_{ext} \equiv Q_1 - Q_{int}$, *or* $Q_2(s) =$

$$\begin{bmatrix} 0 & 0 & \frac{2(s^2+18s+76)}{s^3+21s^2+130s+234} & \frac{s^2+18s+76}{s^3+21s^2+130s+234} \\ 0 & 0 & \frac{2(52+15s+s^2)}{s^3+21s^2+130s+234} & \frac{52+15s+s^2}{s^3+21s^2+130s+234} \\ \frac{2s+13}{s^2+12s+34} & \frac{s+8}{s^2+12s+34} & 0 & 0 \\ \frac{s+10}{s^2+12s+34} & \frac{2(7+s)}{s^2+12s+34} & 0 & 0 \end{bmatrix}$$

$$P_2(s) = \begin{bmatrix} \frac{s^2+19s+86}{s^3+21s^2+130s+234} \\ \frac{(13+s)(s+5)}{s^3+21s^2+130s+234} \\ \frac{7+s}{s^2+12s+34} \\ \frac{s+8}{s^2+12s+34} \end{bmatrix}$$

and taking $Q_{int} \equiv [\mathbf{0}]$ *and* $Q_{ext} \equiv Q_2$. *It is routine to show that with these definitions, both* $(Q_1, P_1)(s)$ *and* $(Q_2, P_2)(s)$ *are consistent with* S_2. *Thus a single subsystem structure can be consistent with two signal structures. It is important to note that one of our signal structures exhibited direct causal dependencies corresponding exactly with the closed loop dependencies described by subsystem structure. We note that this correspondence sometimes occurs (as in the case of Theorem 3) but generally does not hold.*

4.5.4 Impact on Systems Theory

The introduction of partial structure representations suggests new problems in systems theory. These problems explore the relationships between different representations of the same system, thereby characterizing properties of the different representations. For example, classical realization theory considers the situation where a system is specified by a given transfer function, and it explores how to construct a consistent state space description. Many important ideas emerge from the analysis:

1. State realizations are generally more informative than a transfer function representation of a system, as there are typically many state realizations consistent with the same transfer function,

2. Order of the state realization is a sensible measure of complexity of the state representation, and there is a well-defined *minimal* order of any realization consistent with a given transfer function; this minimal order is equal to the Smith-McMillian degree of the transfer function,

3. Ideas of controllability and observability of a state realization characterize important properties of the realization, and any minimal realization is both controllable and observable.

In a similar way, introducing partial structure representations impacts a variety of concepts in systems theory, including realization, minimality, and model reduction.

Realization The definition of partial structure representations enrich the kinds of realization questions one may consider. In the previous section, we demonstrated that partial structure representations of a system are generally more informative than its transfer function but less informative than its state realization. Thus, two classes of representation questions emerge: reconstruction problems and structure realization problems (Figure 30).

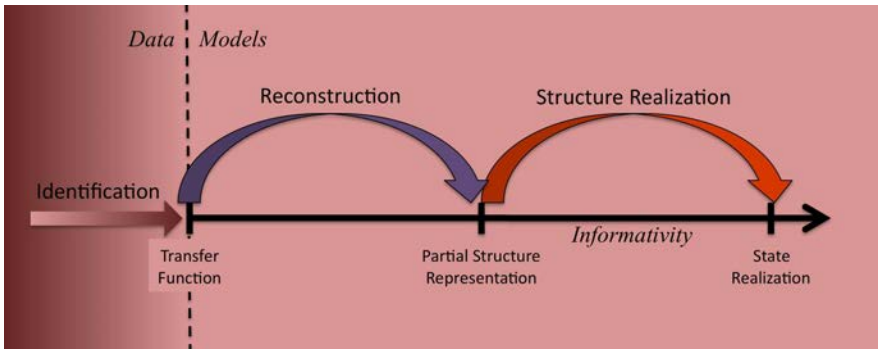


Figure 30: Partial structure representations introduce new classes of realization problems: reconstruction and structure realization. These problems are distinct from identification, and each depends on the type of partial structure representation considered.

Reconstruction problems consider the construction of a partial structure representation of a system given its transfer function. Because partial structure representations are generally more informative than a transfer function, these problems are ill-posed–like the classical realization problem. Nevertheless, one may consider algorithms for generating canonical representations, or one may characterize the additional information about a system–beyond knowledge of its transfer function–that one would require in order to specify one of its partial structure representations. In particular, we may consider two types of reconstruction problems:

1. **Signal Structure Reconstruction:** Given a transfer function $G(s)$ with its associated zero-pattern structure $\mathcal{Z}$, find a signal structure, $\mathcal{W}$, with dynamical structure function (Q, P) as in Equation (60) such that $G = (I - Q)^{-1}P$,

2. **Subsystem Structure Reconstruction:** Given a transfer function $G(s)$ with its associated zero-pattern structure $\mathcal{Z}$, find a subsystem structure, $\mathcal{S}$, with structured LFT (N, S) as in Equation (54) such that $G = (I - SK)^{-1}SL$.

Signal structure reconstruction is also called *network reconstruction*, particularly in systems biology where it plays a central role. There, the objective is to measure fluctuations of various proteins, or other chemical species, in response to particular perturbations of a biochemical system, and then infer causal dependencies among these species [78, 79, 76, 71, 80, 81].

Structure realization problems then consider the construction of a state space model, possibly generalized to include auxiliary variables as necessary, consistent with a given partial structure representation of a system. Like the classical realization problem or reconstruction problems, these problems are also ill-posed since there are typically many state realizations of a given partial structure representation of a system. Also, like reconstruction problems,

these realization problems can be categorized in two distinct classes, depending on the type of partial structure representation that is given:

3. **Signal Structure Realization:** Given a system G with signal structure $\mathcal{W}$ and associated dynamical structure function (Q, P), find a state space model (A, B, C, D) consistent with (Q, P), i.e. such that Equations (57-60) hold,

4. **Subsystem Structure Realization:** Given a system G with subsystem structure $\mathcal{S}$ and associated structured (N, S) (recorded in the LFT representation $\mathcal{F}(N, S)$), find a generalized state space model of the form of Equation (48) consistent with (N, S).

Signal structure realization may sometimes be called *network realization*, consistent with the nomenclature for signal structure reconstruction.

Note that all the reconstruction and structure realization problems here are distinct from identification problems, just as classical realization is different from identification. For the systems considered here, identification refers to the use of input-output data (and no other information about a system) to choose a representation that best describes the data in some appropriate sense. Because input-output data only characterizes the input-output map of a system, identification can at best characterize the system's transfer function; no information about structure, beyond the zero-pattern of the transfer function, is available in such data. In spite of this distinction, however, it is not uncommon for reconstruction problems to be called *structure identification* problems. Nevertheless, one may expose such problems as the concatenation of an identification and a reconstruction problem and precisely characterize the extra information needed to identify such structure by carefully analyzing the relevant reconstruction problem, independent of the particular identification technique [71, 78].

Minimality Just as partial structure representations enrich the classical realization problem, they also impact the way we think about minimality. Certainly the idea of a minimal complexity representation is relevant for each of the four problems listed above, but clearly the relevant notion of complexity may be different depending on the representation. We consider each situation as follows:

1. **Minimal Signal Structure Reconstruction:** In this situation one needs to consider how to measure the complexity of a system's signal structure, $\mathcal{W}$, or its associated dynamical structure function, (Q, P). Some choices one may consider could be the number of edges in $\mathcal{W}$, the maximal order of any element in (Q, P), or the maximal order of any path from any input u_i to any output y_j. The problem would then be to find a minimal complexity signal structure consistent with a given transfer function.

2. **Minimal Subsystem Structure Reconstruction:** In this situation one needs to consider how to measure the complexity of a system's subsystem structure, $\mathcal{S}$, or its associated structured LFT, (N, S). One notion could be to measure complexity by the number of distinct subsystems; the problem would then be to find the minimal complexity subsystem structure representation consistent with a given transfer function. Another notion could be the number of non-zero entries in the S matrix, where (N, S) denote the the LFT associated with the subsystem structure. Using this measure,

a single subsystem block with no zero entries would be considered a more complex representation than a subsystem structure with a large number of distinct, albeit interconnected, subsystems.

3. **Minimal Signal Structure Realization:** In this situation one needs to consider how to measure the complexity of a system's zero-intricacy state realization, from which signal structure is derived. The obvious choice would be to use the order of the realization as a natural measure of complexity, and the problem would then become to find the minimal order state realization consistent with a given signal structure, $\mathcal{W}$, or, equivalently, with its associated dynamical structure function, (Q, P). Note that this minimal order is guaranteed to be finite (for the systems considered here) and can easily be shown to be greater or equal to the Smith-McMillian degree of the transfer function specified by the signal structure; we call this number the *structural degree* of the signal structure [72].

4. **Minimal Subsystem Structure Realization:** In this situation one needs to consider how to measure the complexity of a generalized state realization in the presence of auxiliary variables. Both the order and the intricacy of the realization offer some perspective of its complexity, but one needs to consider how they each contribute to the overall complexity of the realization. The problem would then be to find a minimal complexity generalized state realization consistent with a given subsystem structure.

These various problems demand new ideas for thinking about the complexity of a system's representation, especially that of a partial structure representation. These new ideas about complexity, in turn, introduce opportunities for characterizing minimality of a representation in terms that add insight to our understanding of the relationship between a system's behavior and its structure, much as controllability and observability characterize classical notions of minimality in a system's state realization. Besides suggesting the need for a characterization of minimality, however, these ideas also impact notions of approximation and how we think about model reduction.

Model Reduction Each of the reconstruction and structure realization problems described above have associated with them not only a minimal-representation version of the problem, but also an approximate-representation version of the problem. The minimal-representation versions of these problems, as described above, seek to construct a representation of minimal complexity in the targeted class that is nevertheless consistent with the system description provided. Similarly, approximate-representation versions of these problems seek to construct a representation in the targeted class that has a lower complexity than the minimal complexity necessary to deliver consistency with the system description provided. As a result, consistency with the given system description can not be achieved, so measures of approximation become necessary to sensibly discuss a "best" representation of the specified complexity.

For example, associated with the classical realization problem is the standard model reduction problem. In this situation, a transfer function is specified, and one would like to construct a state realization with a complexity that is lower than that which is necessary (for such a realization to be consistent with the given transfer function) that nevertheless

"best" approximates it. Note that the appropriate notion of complexity depends on the target representation class; here, the target representation class is the set of state realizations, so the appropriate notion of complexity is model order. Likewise, note that the appropriate notion of approximation depends on the type of system representation that is initially provided; here, a transfer function is provided, so an appropriate measure of approximation could be an induced norm, such as H_∞. Thus, one could measure the quality of an approximation by measuring the induced norm of the error between the given transfer function and that specified by the approximate state realization. Note that since the H_∞ model reduction problem remains open, many alternative measures and approaches have been suggested. In any event, because the specified system description is a transfer function, the resulting measure of approximation is typically one that either directly or indirectly measures the difference in input-output dynamic behavior between the approximate model and the given system description; the focus is on dynamics, not system structure, when considering notions of approximation in the standard model reduction problem.

Similarly, there is an appropriate reduction problem associated with each of the minimal reconstruction and minimal structure realization problems described above. Like standard model reduction, the appropriate notion of complexity is characterized by the class of target representations, while the appropriate measure of approximation depends on the system representation that is initially provided. To make these ideas more concrete, we discuss each of the problems individually:

1. **Approximate Signal Structure Reconstruction:** In this situation one would like to find a signal structure representation with lower complexity e.g. fewer number of edges, etc. than the minimal level of complexity necessary to specify a given transfer function. When using the number of edges as a complexity measure, this problem may be interpreted as trying to restrict the number of communication links between individual computation nodes in a distributed system while approximating some desired global dynamic. The appropriate measure of approximation, then, is any measure that sensibly compares the desired transfer function from that specified by the reduced signal structure, such as H_∞.

2. **Approximate Subsystem Structure Reconstruction:** In this situation one would like to find a subsystem structure representation with lower complexity than the minimal level of complexity necessary to specify a given transfer function. If one considers the number of subsystems as the measure of complexity, then this problem is trivial since any transfer function is consistent with a subsystem structure with a single subsystem. If one measures complexity by the number of non-zero entries in the S matrix, where (N, S) denote the the LFT associated with the subsystem structure, then it appears likely that one could often formulate a meaningful approximation problem.

Unlike these approximate reconstruction problems, approximate realization problems may have dual relevant measures of approximation, since having an initial partial structure representation of a system also always specifies its transfer function. Measuring the similarity between transfer functions determines the degree to which a lower order system approximates the *dynamics* of a given system, while measuring the similarity between partial

structure representations determines the degree to which a lower order system approximates the *structure* of a given system.

3. **Approximate Signal Structure Realization:** In this situation one would like to find a state realization with a model complexity that is lower than the minimal complexity necessary to specify a given signal structure. Typically the complexity of a generalized state realization would be measured by both the order and intricacy of the realization. Since signal structure only depends on the zero intricacy realization of a system, however, a minimal complexity realization will always have zero intricacy; thus model order is the only relevant notion of complexity. Moreover, since the structural degree of a particular signal structure may be strictly greater than the Smith-McMillian degree of the transfer function it specifies, a few distinct kinds of approximation problems emerge: structural approximation and dual approximation (Figure 31).

   (a) **Structural Approximation:** When the order of the approximation is less than the structural degree of the given signal structure but greater than the Smith-McMillian degree of the associated transfer function, the resulting approximation can specify the given transfer function exactly, even though its signal structure only approximates that of the given system. Note that the sense in which similarity in signal structure should be measured is an area of on-going research.

   (b) **Dual Approximation:** When the order of the approximation is less than the Smith-McMillian degree of the transfer function specified by the given signal structure, then it will represent neither the structure nor the dynamics of the given system exactly.

4. **Approximate Subsystem Structure Realization:** In this situation one would like to find a generalized state realization with a model complexity that is lower than the minimal complexity necessary to specify a given subsystem structure. Here, complexity of a generalized state realization is measured both in terms of intricacy and order since intricacy of a realization directly impacts the number of admissible subsystem blocks (see Figure 23) while order impacts the ability of the realization to approximate the transfer function specified by the given subsystem structure. As a result, three distinct approximation problems emerge to complement subsystem structure realization: Structure-Preserving Model Reduction, Subsystem Structure Approximation, Subsystem Dual Approximation (Figure 32).

   (a) **Structure-Preserving Model Reduction:** When the intricacy of an approximation is high enough, the subsystem structure of a system can be preserved while its dynamic behavior is approximated by lower-order systems. A naive approach to such reduction would be to reduce the order of various subsystems. However, even when each subsystem is well approximated, the dynamic behavior of the overall interconnection may be very different from the original system. As a result, a rich literature has developed in this area that develops sophisticated techniques for approximating the dynamics of the closed-loop, interconnected system while preserving its subsystem structure [82, 83].

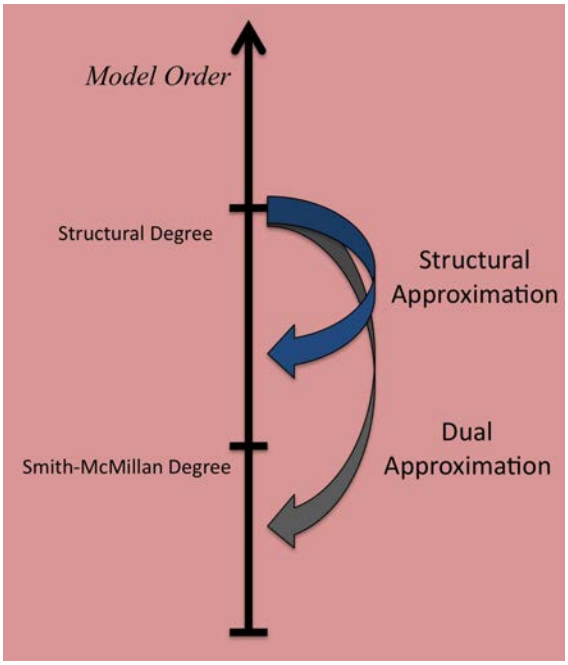


Figure 31: Approximate signal structure realization leads to two distinct types of reduction problems. Structural approximation exactly realizes the dynamics of a given signal structure while only approximating its structure; dual approximation captures neither the dynamics nor the structure of a given signal structure exactly.

(b) **Subsystem Structure Approximation:** When the order of an approximation is high enough, the dynamic behavior of a system can be preserved while is subsystem structure is approximated by lower-intricacy realizations. The sense in which similarity of subsystem structure should be measured remains an open topic for research.

(c) **Subsystem Dual Approximation:** When both the order and the intricacy of an approximation are lower than the minimal values necessary to realize a given subsystem structure and the transfer function it specifies, then the objective of the reduction problem is to find the realization of the specified complexity that best approximates the structure and dynamics of the given system.

The introduction of partial structure representations suggests a number of new problems in systems theory. These problems include new classes of realization problems, called reconstruction and structural realization problems, as well as a number of new reduction problems. Each of these problems differ depending on the partial structure representation one considers, and a number of research issues remain to properly formulate most of them. The overview offered here is merely meant to give a perspective of the landscape of problems that emerges with the introduction of partial structure representations.

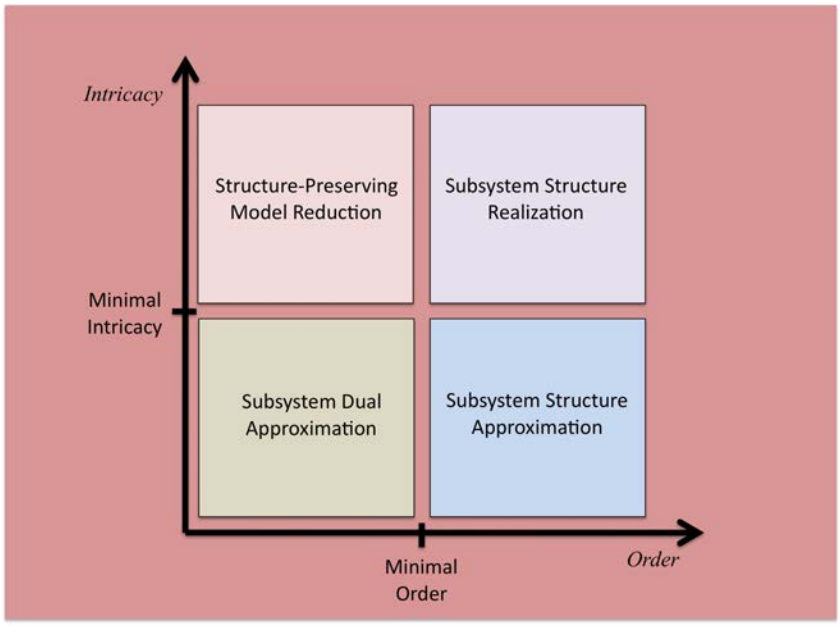


Figure 32: Approximate subsystem structure realization leads to three distinct types of reduction problems: 1) Structure-preserving model reduction preserves the subsystem structure of a given system while approximating its dynamic behavior, 2) Subsystem structure approximation preserves the dynamic behavior but approximates the subsystem structure, and 3) Subsystem dual approximation captures neither the structure nor the dynamic behavior of a system described by a particular subsystem structure.

4.5.5 Summary

This section introduced the idea of a system's complete computational structure as a baseline for comparing simplified representations. Although closely aligned with a system's state space realization, we demonstrated the need for auxiliary variables to encode information about a system's admissible subsystem structure. This results in a "generalized" state description that is a differential algebraic system with differentiation index zero, and the number of auxiliary variables becomes an additional measure of complexity (besides model order) that we call *intricacy*. This generalized state representation, and its associated graphical representation, is the system's complete computational structure.

Partial structure representations were then introduced as a means for simplifying a system's structural description while retaining a complete representation of its input-output dynamic behavior. These included subsystem structure, signal structure, and the zero-pattern of the system's transfer function.

Subsystem structure was first introduced as the most refined view of the interconnection of a system's legitimate subsystems. This description is represented as the lower linear fractional transformation of a static interconnection matrix N with a block diagonal operator S. It's graphical representation is a block diagram, like the system's complete computational structure, that is a condensation graph with respect to a meaningful partition of the system's states.

Signal structure, on the other hand, is a signal flow diagram of the causal dependencies among manifest variables given by the dynamical structure function of the system. We demonstrated that systems may exhibit extremely structured behavior in the signal structure sense while having no apparent structure in the subsystem or computational structure sense. Moreover, the transformations between manifest variables, represented as edges in the signal structure graph, do not necessarily partition the system states, as do the nodes of the subsystem structure. This fact implies that the minimal order to realize a particular signal structure may be, in fact, higher than the minimal order necessary to realize the transfer function specified by the given signal structure.

Finally, the weakest notion of structure is the pattern of zero entries in the system's transfer function matrix, which graphically also becomes a signal flow graph like signal structure. We demonstrated that this representation is very weak, reminding readers that a diagonal transfer function matrix, for example, does not imply that even a minimal realization of the system is necessarily decoupled. Thus, this notion of structure really only makes a statement about the closed-loop dependencies of inputs on outputs of the system.

These representations were then shown to contain differing levels of structural information about a system. In particular, it was shown that the complete computational structure uniquely specifies both the subsystem and signal structure of a system, and that either of these partial structure representations uniquely specify the transfer function (and thus its associated zero-pattern structure) of the system. Nevertheless, the relationship between subsystem and signal structure is less definitive, as we demonstrated that two realizations of the same system may share subsystem structure but have different signal structures, or, conversely, two realization of the same system may share signal structure but have different subsystem structures. These different representations simply appear to offer different perspectives of the system's structural properties.

We then surveyed the landscape of new problems in systems theory that arise when considering these partial structure representations. In particular, we showed that the standard realization problem now becomes two types of problems, a reconstruction problem, where a transfer function is given and one would like to determine a partial structure representation that is consistent with it, and a structure realization problem, where a partial structure representation is given and one would like to find a generalized state realizations that is consistent with it. Minimal versions of these problems are obtained if one can define a sensible notion of complexity of each kind of representation, and various suggestions were offered. Associated approximation, or model reduction problems were then characterized, where the target representation is simpler than the minimal complexity necessary to yield a representation that is consistent with the given model or description of the system. Here we note that the model reduction problems begin to consider structure approximation as well as approximation of the system's dynamic behavior, leading to a variety of new problems one may consider. It is out hope that many of these problems will be addressed in the coming years, leading to a more thorough and complete understanding of the meaning of structure and its relationship to the behavior of interconnected dynamic systems.

4.6 Vulnerable Links and Secure Architectures in the Stabilization of Networks of Controlled Dynamical Systems

The Stuxnet virus attacked an Iranian nuclear power plant in 2010 and caused the centrifuge's rotors to malfunction [98]. It gained much news coverage as the first virus attack on industrial systems. Although no serious damage was done, it has highlighted the necessity of improving our understanding of the security of control systems.

Researchers have predicted that attacks on industrial control systems would increase [86, 84]. As the systems are becoming more networked, securing them has become much harder and attacking them has gotten easier. In the past, securing the physical plants was enough to secure the system, but now in the networked architecture the communication channels have to be secured too. This is almost impossible to achieve, especially when these systems are being connected to the Internet, with connection features as powerful as remote access to the control centers. Regardless of the improvement in the industry's secure communications, cryptography, etc., a simple human error like someone forgetting to change a default password on their account could give an attacker complete access to the resources necessary to carry out a complicated attack. Although the security risks are real, these systems being less networked in the future is highly unlikely because of the usability advantages that a networked setting offers.

As a result, considering the security of networked control systems has become very important. A good design should make detecting attacks easy, help understand the effects of an attack, make it difficult to execute an attack, and finally minimize the consequences if an attack is successful. This section will contribute in designing more secure networked systems by identifying conditions when a link is completely secure against attacks that attempt to destabilize the system. Our result also gives a measure of link vulnerability, which corresponds to the minimum size of a destabilizing attack on the link. This can be a useful tool to understand the security of a networked system.

In this section, we view security as a robustness issue and focus on making systems robust against perturbations on a single communication channel. First, we give a summary of the types of attacks that a system might suffer. Then, in Section 7.6.4 we present our main result. Finally in Section 7.6.5 we give some examples to illustrate the applications of our theory.

4.6.1 Attack Models

In the literature, attacks on control systems have been classified into two types: *denial of service attacks*, when the attacker jams a channel in order to destabilize the system, and *deception attacks*, when the attack adds perturbations on particular links in order to compromises the reliability of the controller's state estimates [96, 85]. We consider a hybrid attack model where the attacker adds perturbations to the channels, not just jam them, in order to destabilize the system. We call this type of attack a *destabilizing attack*.

Denial of Service (DoS) Attack Denial of service attacks prevent signals from reaching their intended destination. This is probably the easiest and most common attack, and it is modeled as removal of an edge in an interconnected structure. It might be done by jamming the communication channel, disrupting the transmitter/receiver, changing the routing protocol, saturating the receiver with extraneous signals, etc. The attacker's intent of such an attack could be to degrade the system performance or to completely destabilize it. [94] shows that performance of networked control systems could decrease significantly under a DoS attack. [96] gives a method to find an optimal controller that minimizes the effect of such an attack on linear control systems.

In [97], the authors study whether a DoS attack on certain links can make the system unreachable or uncontrollable. They also develop graph theoretic algorithms to identify the minimal number of edges which are necessary for preserving controllability and observability.

Deception Attack The goal of a deception attack is to change the state estimates computed by a model-based controller. This type of attack is modeled as a stable additive perturbation to an edge in the network. All stabilizing controllers make the closed loop system stable, hence, a stabilizing controller is necessarily stabilizable from the plant. So, if at attacker gains access to the communication channel between the plant and the controller, state estimates of a model-based controller can be altered. To prevent this, many real systems such as power systems, sensor networks, etc., are equipped with a Bad Data Detector (BDD) [99, 92, 100]. A BDD, using the model of the plant, detects deviation of the state estimates from the expected and raises an alarm to notify the human operator. Because of the presence of measurement noise, this deviation is never zero, so the BDD ignores deviations that are smaller than a specified threshold. Hence, in the presence of BDDs, the attack has to change the state estimates without increasing the chance of raising an alarm.

In [99] the authors study this kind of attack in the context of a power system. They shown that it is in fact possible for an attacker to change the state estimates to a specific value without increasing the chance of being detected. [100] studies a similar problem in the scenario of a wireless sensor networks. It maps an approximation of the set of all possible values the attacker could drive the estimates to.

[92] studies a slightly different problem. Here, the goal of the attacker is to change the estimate of one of the states without increasing the chance of being detected. The authors recognize that while doing this the attacker might want to use the fewest channels possible or might try to keep the magnitude of the attack signal small. For each type of attack, the authors then give a formulation of a *security index* of the system.

Destabilizing Attack Like deception attacks, these attacks effectively arise as an additive perturbation on a link in the system interconnection structure. Unlike deception attacks, however, they seek to destabilize the system rather than simply move the system state to a desired value without being detected. BDDs are clearly capable of detecting the destabilization resulting from such attacks, nevertheless serious damage and even complete plant shut-down may result by the time system operators are able to do anything about it.

A rich literature in systems and control theory explores the destabilization of systems due to additive perturbations, see for example [91] and the references therein. Security analysis of destabilizing attacks thus appears to be a robustness problem with respect to certain classes of perturbations. Indeed, we adopt this point of view, and consider security problems to be essentially robustness problems of various types.

The contribution of this work, then, applied to this class of attacks, is in the solution of a certain class of robustness problems over a particular kind of link model–corresponding to logical, rather than the physical, links of a system–and with respect to a specific class of perturbations. Unlike standard robustness measures that generally consider destabilizing perturbations acting over all channels and nodes of a system, here we restrict our attention specifically to perturbations that disrupt a single link in the system's signal structure. Our analysis then considers such single-link perturbations over all possible system links. In the next section we explore our link model in detail.

4.6.2 Link Models

The destabilizing attacks considered here are additive perturbations acting on a single link in a system's logical interconnection structure. There are many characterizations of a system's structure, see for example [88, 89, 87]. One characterization would consider the interconnection structure among subsystems. This definition of structure, also called the system's subsystem structure, would represent the physical interconnection between physical components of a particular networked system. Under this notion of structure, a *link* would represent the signal passing between two subsystem nodes within the subsystem interconnection architecture. In contrast to the subsystem structure, this work considers another definition of system structure and, consequently, a different notion of a system link.

In this work, we consider a partition on signals of the system into two categories: exposed signals and hidden signals. The logical interconnection structure, or architecture–also called the system's signal structure–is the causal relationship between exposed signals in the system. In this definition of structure, a *link* is a system describing the causal dependency between two exposed signal nodes of the logical interconnection architecture.

Some important consequences of this definition of link include the fact that a link may represent a very indirect and complicated pathway–through various hidden signals that may be components of other links in the system. Thus a link is associated with a particular

set of dynamics–a system–that characterizes how the input signal is transformed into the output signal. The fact that hidden signals may be shard between links, however, is an important distinction between signal and subsystem interconnection structures. Note that a state of one subsystem, interconnected with others in a subsystem architecture (such as a standard feedback interconnection between two blocks), is never shared with other subsystems; the subsystem architecture effectively partitions the states of the networked system. In contrast, states on the links of the signal structure may, in fact, be shared with those of other links. This degree of abstraction is important for security problems because an additive perturbation on a link of the signal structure does not represent the corruption of a particular channel, as it would in the subsystem structure, but rather the idea that an attacker infiltrated a particular dependency between specific manifest variables.

The next section provides some background on Dynamical Structure Functions (DSF), which are used to represent the signal structure of a system. The DSF is a system representation that describes more structure, in the logical interconnection sense, than the transfer function provides, but less than the state space realization would reveal. Specifically, the DSF describes exactly the causal dependencies between manifest variables without offering any indication of the structure relating hidden variables. As a result, although every state space realization specifies a unique DSF, and every DSF specifies a unique transfer function, there are many DSF architectures consistent with any specific transfer function, and many state space realizations consistent with any specific DSF.

4.6.3 Background: Dynamical Structure Function

Before developing the main theorem, we will present a concise derivation of the dynamical structure function, and explain its relevance to the security of a networked system. For a complete derivation and results on different representations of structure see [87, 88].

Let us consider a state-space LTI system

$$\begin{aligned} \begin{bmatrix} \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} &= \begin{bmatrix} \bar{A}_{11} & \bar{A}_{12} \\ \bar{A}_{21} & \bar{A}_{22} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} \bar{B}_1 \\ \bar{B}_2 \end{bmatrix} u \\ y &= \begin{bmatrix} \bar{C}_1 & \bar{C}_2 \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}, \end{aligned} \tag{72}$$

where $\begin{bmatrix} \bar{C}_1 & \bar{C}_2 \end{bmatrix}$ has full row rank. This system can be transformed to:

$$\begin{aligned} \begin{bmatrix} \dot{y} \\ \dot{x} \end{bmatrix} &= \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} y \\ x \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u \\ y &= \begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} y \\ x \end{bmatrix}, \end{aligned} \tag{73}$$

Here y are the states that are measured, and x are the hidden states. Now, taking Laplace Transforms of the signals in (73), we get

$$\begin{bmatrix} sY \\ sX \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} Y \\ X \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} U. \tag{74}$$

Solving for X in the second equation of 74 gives

$$X = (sI - A_{22})^{-1}A_{21}Y + (sI - A_{22})^{-1}B_2U$$

Substituting into the first equation of (74) we get,

$$sY = WY + VU,$$

where $W = A_{11} + A_{12}(sI - A_{22})^{-1}A_{21}$ and $V = A_{12}(sI - A_{22})^{-1}B_2 + B_1$. Let D be a diagonal matrix with the diagonal entries of W. Then,

$$(sI - D)Y = (W - D)Y + VU.$$

Now we can rewrite this equation as,

$$Y = QY + PU, \tag{75}$$

where

$$Q = (sI - D)^{-1}(W - D)$$

and

$$P = (sI - D)^{-1}V.$$

The matrix Q is a matrix of transfer functions from Y_i to Y_j, $i \neq j$, or relating each measured signal to the other measured signals. Note that Q is zero on the diagonal and either zero or a strictly proper transfer function on the off diagonal. The matrix P is a matrix of zeros or strictly proper transfer functions from each input to each output without depending on any additional measured states. Together, the pair $(Q(s), P(s))$ is called the *dynamical structure function* for system (72).

The transfer function matrix for this system is given by

$$G = (I - Q)^{-1}P = C(sI - A)^{-1}B.$$

Hence, G_{ij} is the closed loop transfer function from input j to state i. In this section, we will also refer to the closed loop transfer function between states. A transfer function from state j to state i is represented by H_{ij}, where

$$H = (I - Q)^{-1}.$$

Note that the transfer function from a state to an input is always zero.

Definition 11. *Given a system 72 with signal structure characterized by the dynamical structure function (P,Q), a* link (i, j) *of the system corresponds to any nonzero entry in P or Q.*

Note that P gives the links from the inputs to the measured states, and Q gives the links that represent the dependencies between the measured states. The next section will introduce the notion of vulnerability and characterize vulnerable links in the system's architecture characterized by (P, Q).

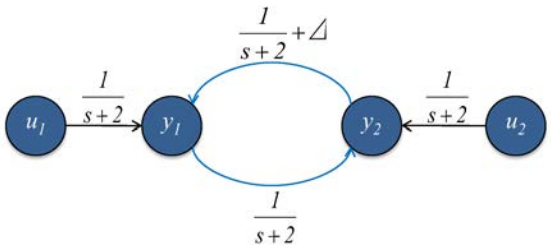


Figure 33: The system with the perturbation Δ. Black arrows indicate secure links, while blue arrows indicate vulnerable links.

4.6.4 Vulnerable Links

In this work, vulnerability refers to the destabilization of a system resulting from the corruption of a single link in its signal architecture. We begin with a definition of a vulnerable link.

Definition 12. *Given a system 72 with signal structure characterized by the dynamical structure function (P,Q), a link in (P,Q) is called* vulnerable *if there exists a stable perturbation on the link that makes the system unstable.*

Example 7. *Let us consider a system with*

$$P = \begin{bmatrix} \frac{1}{s+2} & 0 \\ 0 & \frac{1}{s+2} \end{bmatrix}, \text{ and } Q = \begin{bmatrix} 0 & \frac{1}{s+2} \\ \frac{1}{s+2} & 0 \end{bmatrix}.$$

This system is stable because the transfer function,

$$G = \frac{1}{s^2+4s+3}\begin{bmatrix} s+2 & 1 \\ 1 & s+2 \end{bmatrix}$$

Now let us add a perturbation $\Delta = \frac{3}{s+2}$ to the link Q_{12} as shown in Figure 33. The resulting transfer function is

$$\bar{G} = \frac{1}{s(s+4)}\begin{bmatrix} s+2 & 1 \\ 4 & s+2 \end{bmatrix},$$

which is unstable. Hence the link Q_{12} is a vulnerable link. Similarly, it can be shown that Q_{21} is vulnerable, although neither P_{11} nor P_{22} are vulnerable.

Condition for Vulnerability Given that an attacker has the knowledge of the dynamical structure function representation of a system, we will derive a necessary and sufficient condition for a link to be vulnerable.

Theorem 5. *Let us consider a stable system (P, Q). There exists a stable additive perturbation Δ on a link from node i to node j, either in P or Q, that makes the system unstable if and only if the closed loop transfer function from node j to i is nonzero.*

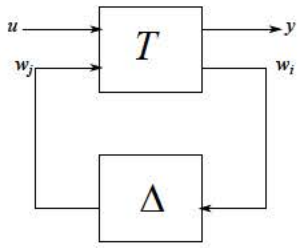


Figure 34: System with the perturbation $\Delta e_i e_j^T$

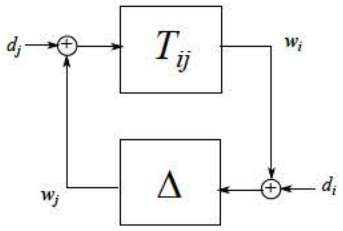


Figure 35: Necessary and sufficient condition for stability of the system in Figure 34

Proof. The system with the perturbation Δ can be represented as the linear fractional transformation in Figure 34, where T is the associated closed loop transfer function, and w_i, w_j represent the signals at node i and j respectively. This system is stable if and only if the system in Figure 35 is stable (see [91]). If $T_{ij} = 0$, any stable Δ does not affect the stability of the system in Figure 35. Thus the closed loop system in Figure 34 is stable for all Δ.

If $T_{ij} \neq 0$, then the system in Figure 35 is unstable if any of the transfer functions of $\begin{bmatrix} d_j \\ d_i \end{bmatrix} \rightarrow \begin{bmatrix} w_j \\ w_i \end{bmatrix}$ is unstable. We have,

$$w_j = \frac{1}{1 - T_{ij}\Delta} \begin{bmatrix} T_{ij}\Delta & \Delta \end{bmatrix} \begin{bmatrix} d_j \\ d_i \end{bmatrix}.$$

Let $T_{ij} = \frac{N}{D}$ and $\Delta = \frac{\delta_N}{\delta_D}$, then

$$w_j = \frac{D\delta_D}{D\delta_D - N\delta_N} \begin{bmatrix} \frac{N\delta_N}{D\delta_D} & \frac{\delta_N}{\delta_D} \end{bmatrix} \begin{bmatrix} d_j \\ d_i \end{bmatrix}. \tag{76}$$

For a polynomial to be stable it is necessary that all its coefficients are of the same sign. In the case of the polynomial

$$R(s) = D\delta_D - N\delta_N, \tag{77}$$

it is easy to see that a properly designed Δ can zero out at least one of the terms. Thus, there exists a Δ that destabilizes these transfer functions. □

Note that when we are considering the vulnerability of the links in Q, $T = H = (I - Q)^{-1}$, gives the closed loop transfer functions. Now, we will present some implications of this result.

Corollary 2. *None of the links in P are vulnerable.*

Proof. This is true because the transfer function from the states to the input is always zero. □

Corollary 3. *If T_{ij} is nonzero, there exists a perturbation $\Delta \in \mathbb{R}$ that destabilizes the system in Figure 35.*

Proof. Let $\Delta \in \mathbb{R}$, $l_{ij} = \frac{N_l}{D_l}$. Thus, $\frac{\delta_n}{\delta_d} = \frac{\Delta D_l + N_l}{D_l}$, and the polynomial in (77) becomes $D_l D - N(D_l \Delta + N_l)$. We can see that at least one of the terms in this polynomial can be zeroed out by choosing appropriate Δ, making the polynomial unstable. □

Corollary 4. *Let us consider a stable system,*

$$\begin{aligned} \dot{x} &= Ax + Iu, \\ y &= Ix, \end{aligned} \tag{78}$$

where $A \in \mathbb{R}^{n \times n}$ and let $G = (sI - A)^{-1}$. There exists a perturbation $K = \Delta e_i e_j^T, \Delta \in \mathbb{R}$, such that $(A+K)$ is not Hurwitz, if and only if the transfer function from input u_i to output y_j, G_{ji}, is nonzero.

Proof. If the perturbation is on the diagonal entry of A, then it is easy to see that a destabilizing perturbation always exists and G_{ii} is never zero. Let $D = diag(A_{11}, A_{22}, ..., A_{nn})$. The dynamical structure function of the system is given by $P = (sI - D)^{-1}$ and $Q = (sI - D)^{-1}(A - D)$. Any perturbation $K = \Delta e_i e_j^T, i \neq j$ effects only the link Q_{ij}. Hence, the perturbation can make the system unstable if and only if the transfer function H_{ji} is nonzero. Also, the diagonal entries of P are nonzero, and $G = HP$. Thus, the transfer function H_{ij} is nonzero if and only if G_{ji} is nonzero.

□

Example 8. *Let us consider a system of the form 78 with*

$$A = \begin{bmatrix} -1 & 0 & -4 & 3 \\ 2 & -2 & 0 & 0 \\ 3 & 0 & -2 & -4 \\ 0 & 3 & -2 & -5 \end{bmatrix}.$$

Here the eigenvalue of A are $\sigma = \{-1.5000 + 3.4278j, -1.5000 - 3.4278j, -6.7016, -0.2984\}$. Hence, the system is stable. In this system, the link from x_4 to x_1 is not vulnerable because $G_{41} = 0$. Notice that this example is not a trivial example, like a diagonal or a triangular system, since there are cycles that contain both nodes x_1 and x_4.

Corollary 5. *Let $A \in \mathcal{R}^{n \times n}$. A perturbation on the $(i,j)^{th}$ entry of A changes its eigenvalues if and only if the $G_{ji} \neq 0$, where $G = (sI - A)^{-1}$ is the transfer function matrix i.e. the (i,j) minor of $(sI - A)$ is nonzero.*

Proof. Take the system from Corollary 4. We can see that a perturbation on the $(i,j)^{th}$ entry has no effect on the system if $G_{ij} = 0$. Also, if $G_{ji} \neq 0$, the perturbation forms a closed loop system, such as the one given in Figure 35, in which case Δ definitely changes the poles of the system. □

If we take the A matrix from Example 8, note that its eigenvalues stay unchanged for any perturbation on the $(1,4)^{th}$ entry.

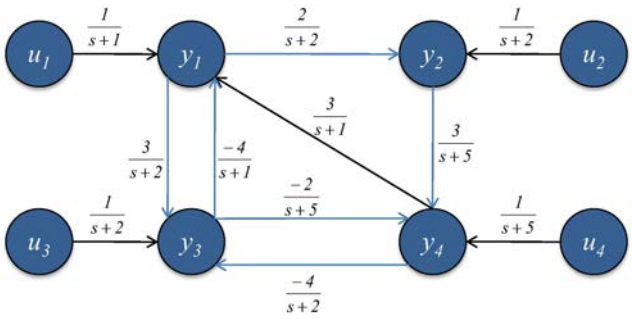


Figure 36: A system with a secure link in a cycle. Black arrows represent the secure links.

Structure and Vulnerability To perform the vulnerability analysis of a system, we assume that the attacker can only modify existing links and cannot create new links. With this assumption, we can see that systems where the output nodes do not form a cycle are always secure, because in such a case the nodes can be permuted to obtain a triangular Q matrix. A triangular Q matrix gives a triangular H, and by applying Theorem 5 we can see that all the existing links are secure. Note that secure links doesn't always mean they are from a triangular system. For example, the link Q_{14} is secure in the system given in Figure 36, which is the signal structure architecture of the state-space system in Example 8.

Noting that certain graphical structures result in secure links begs the question of whether there are particular dynamics that contribute to secure or vulnerable links in the system's architecture. The following theorem answers this question.

Theorem 6. *Every transfer function G has a completely secure architecture $(\bar{P}, \bar{Q})$.*

Proof. For any transfer function G, note that $(P = G, Q = 0)$ is an admissible Dynamical Structure Function since $G = (I - 0)^{-1}G$. From Corollary 1, we see that none of the links in P are vulnerable, and since Q has no links, the system is secure. □

This result shows that the vulnerability of a system is structure dependent and not a function of the system dynamics. This fact highlights one difference between the vulnerability, which depends on the system structure and not the dynamics, and the robustness, which depends on the dynamics and not the system structure.

Measure of Vulnerability Feedback is very common in both natural and engineered systems. Nevertheless, such structures usually generate vulnerable links. Thus, a measure of vulnerability is essential to understand the security of the system.

Given a signal architecture (P, Q) with associated closed loop transfer function T, the vulnerability of link (i, j) is given by

$$v_{ji} = \frac{1}{||T_{ij}||_\infty}, \tag{79}$$

which is the smallest perturbation required on link (i, j) to destabilize the system. Since all the links in P are secure, we only consider the links in Q while computing the vulnerability,

hence, $T = H$. The vulnerability of the system is given by

$$V = \min_{(i,j)\in Q} v_{ji} \tag{80}$$

$$= \min_{(i,j)\in Q} \frac{1}{||T_{ij}||_\infty} \tag{81}$$

This measure allows us to associate a size of the smallest destabilizing perturbation with every link in the system architecture. Secure links thus have a vulnerability of ∞. Note that V, the system vulnerability, is different than (and not smaller than) the size of the smallest destabilizing perturbation for the system, since link perturbations are restricted to act on a single link only.

4.6.5 Numerical Example

Let us consider a system with the architecture given in Figure 37a where,

$$P = \begin{bmatrix} \frac{1}{s+1} & 0 & 0 \\ 0 & \frac{1}{s+1} & 0 \\ 0 & 0 & \frac{1}{s+1} \end{bmatrix}$$

and

$$Q = \begin{bmatrix} 0 & 0 & \frac{1}{s+1} \\ \frac{1}{s+2} & 0 & 0 \\ 0 & \frac{1}{s+3} & 0 \end{bmatrix}.$$

The transfer function matrix for the system is given by

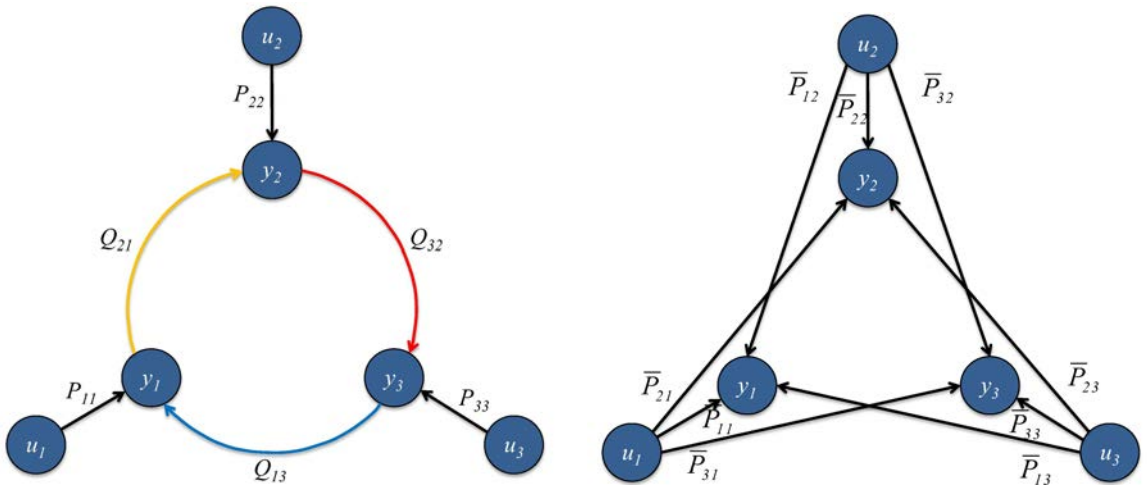
$$G = \begin{bmatrix} \frac{s^3+6s^2+11s+6}{d(s)} & \frac{s+2}{d(s)} & \frac{s^2+5s+6}{d(s)} \\ \frac{s^2+4s+3}{d(s)} & \frac{s^3+6s^2+11s+6}{d(s)} & \frac{s+3}{d(s)} \\ \frac{s+1}{d(s)} & \frac{s^2+3s+2}{d(s)} & \frac{s^3+6s^2+11s+6}{d(s)} \end{bmatrix},$$

where $d(s) = s^4 + 7s^3 + 17s^2 + 16s + 5$. The gain of this system is given by $||G||_\infty = 2.4$. Hence, by small gain theorem, a perturbation of gain larger than 0.42 could destabilize the system.

Let $H = (I - Q)^{-1}$ represent the transfer function between the measured states y_i. Since the links in P are not vulnerable, we consider the perturbations on the links in Q which are the links (y_1, y_2), (y_2, y_3), and (y_3, y_1). To compute the vulnerability of these links we need the following transfer functions:

$$H_{12} = \frac{s+2}{s^3+6s^2+11s+5}$$
$$H_{23} = \frac{s+3}{s^3+6s^2+11s+5}$$
$$H_{31} = \frac{s+1}{s^3+6s^2+11s+5}.$$

For this system $v_{12} = \Delta_{12} = 2.5$, $v_{23} = \Delta_{23} = \frac{5}{3}$, and $v_{31} = \Delta_{31} = 5$. Hence, the smallest perturbation on a single link that can destabilize this system must have a gain of 1.67.



(a) Vulnerable architecture (b) Completely secure architecture

Figure 37: Vulnerable and secure architectures for the same transfer function. Black links are secure, vulnerable links are colored blue, yellow, and red in the increasing order of their vulnerability.

This system can also be implemented as shown in Figure 37b, where $\bar{P} = G$. This is one of the secure implementations of the system in Figure 37a. From this example we thus observe the following:

- The same transfer function can exhibit both vulnerable and secure architectures,
- System robustness, characterized by the size of the smallest destabilizing perturbation (0.42 in this example), is not equivalent to system vulnerability, characterized by the size of the smallest destabilizing perturbation on a single link (about 1.67 in this example),
- Only links in Q can be vulnerable.

4.6.6 Summary

This section explored the notion of a vulnerable link in a network of controlled linear dynamical systems. The architecture of the system was characterized by its Dynamical Structure Function, representing the logical interconnection structure of the system. Vunerability was then defined as the size of the smallest destabilizing perturbation acting on a single link. The main results of the section provided necessary and sufficient conditions for the vulnerability of a link and then demonstrated that any transfer function has a completely secure architecture. This result highlights the idea that while robustness is a property of a system's dynamics, security (in the sense discussed here) is a property of its signal architecture. Future work will explore parallel notions for subsystem interconnection structure of networked systems.

5 Conclusions

This work considered a new model for complex network environments and explored its applicability to wireless mesh networks and bio-chemical reaction networks. The new model characterizes structure of a networked system in a novel way that does not attempt to partition every state variable of the system into well defined subsystems, resulting in a significant reduction in the information load necessary to reconstruct the (new) network structure from experimental data. Instead, the new model characterizes the structure of the network as the dependency among manifest variables (i.e. those variables that are visible to the outside world), and this structure can be discovered with O(n) experiments, where n is the number of manifest variables (observed nodes). Moreover, algorithms for network reconstruction were developed that apply to nonlinear systems near equilibrium with noisy measurements. These "robust" reconstruction methods have been tested in-silico, and experiments to validate them on both wireless mesh and bio-chemical reaction networks remain a goal for future research.

Although the new network model can be applied directly to existing models of chemical reaction kinetics, application to wireless mesh networks is less obvious. As a result, significant efforts were made to model and understand the behavior of wireless mesh networks leveraging the team's varied expertise through the "network design cycle." Following this process, we introduced a new partial interference model and a new first principles model for wireless mesh networks, used them to design novel rate control protocols, and then verified the improved performance of these protocols through both simulation and experimental testing. Although these models are not yet exactly the type of model needed for network reconstruction, they lay the appropriate foundation for developing such a model. Our next steps will be to extend this work to a model appropriate for network reconstruction in wireless mesh networks, and then use this model to experimentally validate our reconstruction methods for these networked systems.

The new network reconstruction model also enables a rigorous analysis of network vulnerabilities. Preliminary work has demonstrated that certain network links can exhibit system-wide vulnerability, and corresponding analysis has demonstrated that every system has a completely secure architecture. In practice this architecture may be unrealizable, since feedback is often an essential part of networked systems, but these early results point to the power of the new models in enabling a rigorous vulnerability analysis by exploring the robustness of the closed-loop system. Future work will build on these results and continue to explore security implications of network attacks. Ultimately, for example, we would like to be able to exploit our network reconstruction technologies to detect intruders in our wireless mesh networks and understand how to deploy specific topologies that minimizes security risks of link-wise attacks.

References

[1] The netfilter framework and iptables homepage.

[2] B. Bensaou and Zuyuan Fang. A Fair MAC Protocol for IEEE 802.11-Based Ad Hoc Networks: Design and Implementation. *IEEE Transactions on Wireless Communications*, 6(8):2934–2941, 2007.

[3] Coen Bron and Joep Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

[4] Randy Buck, Rich Lee, Phil Lundrigan, Charles Knutson, and Daniel Zappala. WiFu: A software toolkit for experimental wireless transport protocols. In preparation.

[5] Lijun Chen, S.H. Low, and J.C. Doyle. Joint congestion control and media access control design for ad hoc wireless networks. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 3, pages 2212–2222 vol. 3, March 2005.

[6] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle. Layering as optimization decomposition: A mathematical theory of network architectures. *Proceedings of the IEEE*, 95(1):255–312, 2007.

[7] Norishige Chiba and Takao Nishizeki. Arboricity and subgraph listing algorithms. *SIAM Journal on Computing*, 14:210–223, 1985.

[8] Saumitra M. Das, Dimitrios Koutsonikolas, Y. Charlie Hu, and Dimitrios Peroulis. Characterizing multi-way interference in wireless mesh networks. In *Proceedings of the 1st international workshop on Wireless network testbeds, experimental evaluation & characterization*, WiNTECH '06, pages 57–64, New York, NY, USA, 2006. ACM.

[9] Xia Gao, Vaduvur Bharghavan, Tae-Eun Kim, and Thyagarajan Nandagopal. Achieving MAC layer fairness in wireless packet networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, pages 87–98, New York, NY, USA, 2000. ACM.

[10] Kamal Jain, Jitendra Padhye, Venkata N. Padmanabhan, and Lili Qiu. Impact of interference on multi-hop wireless network performance. *Wirel. Netw.*, 11(4):471–487, 2005.

[11] Anand Kashyap, Samrat Ganguly, and Samir R. Das. A measurement-based approach to modeling link capacity in 802.11-based wireless networks. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, MobiCom '07, pages 242–253, New York, NY, USA, 2007. ACM.

[12] Yi Li, Lili Qiu, Yin Zhang, Ratul Mahajan, and Eric Rozner. Predictable performance optimization for wireless networks. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, SIGCOMM '08, pages 413–426, New York, NY, USA, 2008. ACM.

[13] S.H. Low and D.E. Lapsley. Optimization flow control. i. basic algorithm and convergence. *Networking, IEEE/ACM Transactions on*, 7(6):861–874, Dec. 1999.

[14] Dragoş Niculescu. Interference map for 802.11 networks. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 339–350, New York, NY, USA, 2007. ACM.

[15] Jitendra Padhye, Sharad Agarwal, Venkata N. Padmanabhan, Lili Qiu, Ananth Rao, and Brian Zill. Estimation of link interference in static multi-hop wireless networks. In *Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, IMC '05, pages 28–28, Berkeley, CA, USA, 2005. USENIX Association.

[16] Lili Qiu, Yin Zhang, Feng Wang, Mi Kyung Han, and Ratul Mahajan. A general model of wireless interference. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, MobiCom '07, pages 171–182, New York, NY, USA, 2007. ACM.

[17] Charles Reis, Ratul Mahajan, Maya Rodrig, David Wetherall, and John Zahorjan. Measurement-based models of delivery and interference in static wireless networks. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '06, pages 51–62, New York, NY, USA, 2006. ACM.

[18] M Bansal, V Belcastro, A Ambesi-Impiombato and D Bernardo, "How to infer gene networks from expression profiles", Molecular Systems Biology, 3:78, 2007.

[19] M Bansal and D di Bernardo, "Inference of gene networks from temporal gene expression profiles", IET Syst. Biol., 1(5):306312, 2007.

[20] D di Bernardo, M Thompson, T Gardner, S Chobot, E Eastwood, A Wojtovich, S Elliott, S Schaus and J Collins, "Chemogenomic profiling on a genome-wide scale using reverse-engineering gene networks", Nature Biotechnology, 23(3), 2005.

[21] K Basso, A Margolin, G Stolovitzky, U Klein, R Dalla-Favera and A Califano, "Reverse engineering of regulatory networks in human B cells", Nature Genetics, 37(4):382–390, 2005

[22] R Bonneau, D Reiss, P Shannon, M Facciotti, L Hood, N Baliga and V Thorsson, "The inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets *de novo*", Genome Biology 7:R36, 2006.

[23] K Burnham and D Anderson, *Model Selection and Inference - A practical information-theoretic approach.* Springer-Verlag, 1998.

[24] A Butte and I Kohane, "Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements", Pac. Symp. Biocomput., pp. 418429, 2000.

[25] I Cantone, L Marucci, F Iorio, M Rucci, V Belcastro, M Bansal, S Santini, M Bernardo, D Bernardo and M Cosma, "A yeast synthetic network for *in vivo* assessment of reverse-engineering and modeling approaches", Cell, 137:172–181, 2009.

[26] N Dojer, A Gambin, A Mizera, B Wilczynski and J Tiuryn, "Applying dynamic Bayesian networks to perturbed gene expression data", BMC Bioinformatics, 7, 2006

[27] J Faith, B Hayete, J Thaden, I Mogno, J Wierzbowski, G Cottarel, S Kasif, J Collins and T Gardner, "Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles", PLoS Biol., 5(1), 2007.

[28] T Gardner, D Bernardo, D Lorenz and J Colins, "Inferring genetic networks and identifying compound mode of action via expression profiling", Science, 301, 2003.

[29] J Gonçalves and S Warnick, "Necessary and sufficient conditions for dynamical structure reconstruction of LTI networks", IEEE Transactions on Automatic Control, vol. 53, 2008.

[30] M Hecker, S Lambeck, S Toepfer, E Someren and R Guthke, "Gene regulatory network inference: data integration in dynamic models–a review", BioSystems, 2009.

[31] A Hirotugu, "A new look at the statistical model identification," IEEE Transactions on Automatic Control 19 (6): 716-723, 1974.

[32] M Koyuturk, "Algorithmic and analytical methods in network biology", WIREs Systems Biology & Medicine, 2(3):277-292, 2010.

[33] L. Ljung. *System Identification–Theory for the User*. Prentice Hall, 1999.

[34] T Nordling and E Jacobsen, "Interampatteness–a generic property of biochemical networks." IET Syst Biol, 2009.

[35] M Roberts, E August, A Hamadeh, P Maini, P McSharry, J Armitage and A Papachristodoulou, "A model invalidation-based approach for elucidating biological signalling pathways, applied to the chemotaxis pathway in R. sphaeroides", BMC Systems Biology, vol. 3., issue 105, 2009.

[36] R De Smet and K Marchal, "Advantages and limitations of current network inference methods", Nature Reviews, Microbiology, 8, 2010.

[37] E Sontag, "Network reconstruction based on steady-state data", Essays in Biochemistry, 45:161-176, 2008.

[38] G Wadhams and J Armitage, "Making sense of it all: Bacterial chemotaxis," Nat Rev Mol Cell Biol 2004, 5:10241037.

[39] N Young, *An introduction to Hilbert space.* Cambridge university press, 1988.

[40] J Yu, V Smith, P Wang, A Hartemink and E Jarvis, "Advances to bayesian network inference for generating causal networks from observational biological data", Bioinformatics, 20(18):3594–3603, 2004.

[41] Y Yuan, G Stan, S Warnick and J Goncalves, "Minimal dynamical structure realisations with application to network reconstruction from data," Proceedings of Conference on Decision and Control, 2009.

[42] Y Yuan, G Stan, S Warnick and J Goncalves, "Robust dynamical network reconstruction from noisy data," Proceedings of Conference on Decision and Control, 2010.

[43] K Zhou, J Doyle and K Glover, "*Robust and Optimal Control*," Prentice Hall, 1996.

[44] J Gonçalves and S Warnick, "*System Theoretic Approaches to Network Reconstruction*," Control Theory and Systems Biology, MIT Press, Eds. B. Ingalls and P. Iglesias, 2009.

[45] E. Yeung, J. Goncalves, H. Sandberg, and S. Warnick. "The Meaning of Structure in Interconnected Dynamic Systems," to appear, IEEE Control Systems Magazine, Special Issue on Designing Controls for Modern Infrastructure Networks, 2011.

[46] Y. Yuan, G. Stan, S. Warnick and J. Gonçavles, "Robust Dynamical Network Reconstruction from Data," Special Issue on System Biology, Automatica, 47(6): 1230-1235, 2011.

[47] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control-Analysis and Design.* Wiley,1996.

[48] E. G. Gilbert, "Controllability and Observability in Multivariable Control Systems," J.S.I.A.M. Control Ser. A, 1: 128-151, 1963.

[49] F. Fallside, *Control System Design by Pole-Zero Assignment.* Academic Press INC. LTD.

[50] C. Godsil and G. Royal, *Algebraic Graph Theory.* New York: Springer-Verlag, 2001.

[51] I. Bomze, M. Budinich, P. Pardalos, M. Pelillo, "The Maximum Clique Problem", Handbook of Combinatorial Optimization, 1999.

[52] J. Robson, "Finding a Maximum Independent Set," J. Algorithm, 7:425-440, 1986.

[53] H. Nyquist, "Regeneration theory", *Bell System Technical Journal*, vol. 11, pp. 126–147, September 1932.

[54] H. S. Black, "Stabilized feedback amplifiers", *Bell System Technical Journal*, vol. 13, pp. 1–18, 1934.

[55] H. W. Bode, "Relations between attenuation and phase in feedback amplifier design", *Bell System Technical Journal*, vol. 19, pp. 421–454, 1940.

[56] J. C. Maxwell, "On governors", *Proceedings of the Royal Society*, vol. 16, no. 100, pp. 270–283, 1868.

[57] H. Raza and P. Ioannou, "Vehicle following control design for automated highway systems", *Control Systems Magazine*, vol. 16, no. 6, December 1996.

[58] J. Fowler and R. D'Andrea, "A formation flight experiment", *Control Systems Magazine*, October 2003.

[59] R. D'Andrea and G. Dullerud, "Distributed control for spatially interconnected systems", *IEEE Transactions on Automatic Control*, vol. 48, no. 9, pp. 1478–1496, September 2003.

[60] D. Del Vecchio, A. J. Ninfa, and E. D. Sontag, "Modular cell biology: retroactivity and insulation", *Molecular Systems Biology*, vol. 4, no. 161, 2008.

[61] S. Jayanthi and D. Del Vecchio, "On the compromise between retroactivity attenuation and noise amplification in gene regulatory networks", *Proceedings of the 2009 IEEE Conference on Decision and Control*, December 2009.

[62] D. Luenberger, "Time-invariant descriptor systems", *Automatica*, vol. 14, pp. 473–480, 1978.

[63] David G. Luenberger, *Introduction to Dynamic Systems : Theory, Models, and Applications*, Wiley, N.Y., 1992.

[64] D. Luenberger, "Dynamic equations in descriptor form", *IEEE Transactions on Automatic Control*, vol. 22, no. 3, pp. 312–321, June 1977.

[65] K. Schmidt, *Dynamical Systems of Algebraic Origin*, Basel, Boston, 1943.

[66] D. Siljac, *Large Scale Dynamic Systems: Stability and Structure*, New York: North-Holland, 1978.

[67] H.S. Mortveit and C.M. Reidys, "Discrete, sequential dynamical systems", *Discrete Mathematics*, vol. 226, pp. 281–295, 2001.

[68] H.S. Mortveit and C.M. Reidys, "Elements of a theory of simulation ii: Sequential dynamical systems", *Applied Mathematics and Computation*, vol. 107, pp. 121–136, 2001.

[69] F. Harary, *Graph Theory*, Addison-Wesley Pub. Co., Reading, Massachusetts, 1969.

[70] K. Zhou J. Doyle, K. Glover, *Robust and Optimal Control*, Prentice Hall, Englewood Cliffs, N.J., 1996.

[71] J. Gonçalves, R. Howes, and S. Warnick, "Dynamical structure functions for the reverse engineering of LTI networks", *IEEE Transactions of Automatic Control, 2007*, August 2007.

[72] E. Yeung, J. Gonçalves, H. Sandberg, and S. Warnick, "Network structure preserving model reduction with weak a priori structural information", *Proceedings of 2009 Conference on Decision and Control*, December 2009.

[73] E. Yeung, J. Gonçalves, H. Sandberg, and S. Warnick, "Representing structure in interconnected lti dynamic systems", *submitted to the Proceedings of IEEE Conference on Decision and Control*, December 2010.

[74] R. Howes, S. Warnick, and J. Gonçalves, "Dynamical structure analysis of sparsity and minimality heuristics for reconstruction of biochemical networks", *Proceedings of the Conference on Decision and Control*, December 2008.

[75] J. Gonçalves, R. Howes, and S. Warnick, "Dynamical structure functions for the reverse engineering of LTI networks", *Proceedings of the Conference on Decision and Control*, 2007.

[76] Yuan Y., Stan G.B., Warnick S., and Gonalves J., "Minimal dynamical structure realisations with application to network reconstruction from data", in *Proceedings of the 48th IEEE Conference on Decision and Control (IEEE-CDC 2009)*, December 2009.

[77] C. Ward, E. Yeung, T. Brown, B. Durtschi, S. Weyerman, R. Howes, J. Goncalves, H. Sandberg, and S. Warnick., "A comparison of network reconstruction methods for chemical reaction networks", *Proceedings of the Foundations for Systems Biology and Engineering Conference*, August 2009.

[78] S. Warnick and J. Gonçalves, *Systems Theoretic Approaches to Network Reconstruction*, chapter 13, pp. 265–296, MIT Press, Cambridge, Massachusetts, 2009.

[79] Claudia Rangel, John Angus, Zoubin Ghahramani, and David L. Wild, *Modeling Genetic Regulatory Networks using Gene Expression Profiling and State-Space Models*, chapter 9, pp. 269–293, Springer, London, UK, 2005.

[80] Warnick S. Yuan Y., Stan G. and Goncalves J., "Robust dynamical network reconstruction", in *19th International Symposium on Mathematical Theory of Networks and Systems (MTNS 2010), 2010*, 2010.

[81] Stan G. Yuan Y. and Goncalves J., "Biological network reconstruction from data. (for engineer)", in *Engineering principle in Biology*. Cold Spring Habour Lab, 2009.

[82] Henrik Sandberg and Richard M. Murray, "Model reduction of interconnected linear systems", *Optimal Control Application and Methods*, 2009, Preprint at `http://www.ee.kth.se/~hsan/intermodred.pdf`.

[83] Henrik Sandberg and Richard M. Murray, "Model reduction of interconnected linear systems using structured gramians", in *Proceedings of the 17th IFAC World Congress*, Seoul, Korea, July 2008, pp. 8725–8730.

[84] A. A. Cardenas, S. Amin, and S. Sastry, "Research challenges for the security of control systems," in the Proceedings of the 3rd conference on Hot topics in security, pages 16, Berkeley, CA, USA, 2008.

[85] A. A. Cardenas, S. Amin, and S. Sastry, "Secure control: Towards survivable cyber-physical systems," in Distributed Computing Systems Workshops, 2008. ICDCS 08. 28th International Conference on, pages 495500, June 2008.

[86] E. Byres and J. Lowe, "The myths and facts behind cyber security risks for industrial control systems," in the Proceedings of the VDE Kongress, VDE Congress, 2004.

[87] E.Yeung, J. Goncalves, H. Sandberg, S. Warnick, "Mathematical relationships between representations of structure in linear interconnected dynamical systems," to appear in the Proceedings of the American Control Conference, 2011.

[88] E. Yeung, J. Goncalves, H. Sandberg, S. Warnick, "The meaning of structure in interconnected dynamic systems," submitted to a special edition of Controls Systems Magazine on Control of Infrastructure Networks.

[89] E. Yeung, J. Goncalves, H. Sandberg, S. Warnick, "Representing Structure in Linear Interconnected Dynamical Systems," in the Proceedings of IEEE Conference on Decision and Control, Atlanta, USA, December, 2010.

[90] E. Yeung. "Network Modeling in LTI dynamical systems: reconstruction, reduction, representation.'. Undergraduate Honors Thesis, Department of Computer Science, Brigham Young University, 2010.

[91] G. E. Dullerud and F. Paganini. *A course in robust control theory, a convex approach*,' Springer, 2000.

[92] H. Sandberg, A. Teixeira, and K. H. Johansson, On security indices for state estimators in power networks," in Preprints of the First Workshop on Secure Control Systems, CPSWEEK, 2010.

[93] J. Goncalves, S. Warnick. "Necessary and Sufficient Conditions for Dynamical Structure Reconstruction of LTI Networks," IEEE Transactions on Automatic Control, August 2008.

[94] M. Long, C.H. Wu, J. Y. Hung, "Denial of Service Attacks on Network-Based Control Systems: Impact and Mitigation," in IEEE Transactions on Industrial Informatics, vol 1, no. 2, May 2005.

[95] R. Howes. "Application and Properties of Dynamical Structure Functions". Undergraduate Honors Thesis, Department of Computer Science, Brigham Young University, 2008.

[96] S. Amin, A. Cardenas, and S. S. Sastry, "Safe and secure networked control systems under denial-of-service attacks," in Hybrid Systems: Computation and Control, pages 3145. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, April 2009.

[97] V. Pichai, M.E. Sezer, D.D. Siljak, "Vulnerability of dynamic systems," in International Journal of Control, 1981, pp. 1049-1060.

[98] Wikipedia, *Stuxnet*. `http://en.wikipedia.org/wiki/Stuxnet`, retrieved 3/17/2011.

[99] Y. Liu, P. Ning, and M. Reiter, "False data injection attacks against state estimation in electric power grids," in the Proceedings of the 16th ACM conference on Computer and communications security, Chicago, Illinois, 2009, pp. 2132.

[100] Y. Mo, E. Garone, A. Casavola, B. Sinopoli, "False data injection attacks against state estimation in wireless sensor networks," in the Proceedings of 49th IEEE Conference on Decision and Control, Atlanta, GA, USA, 2010.